



TAMPEREEN TEKNILLINEN YLIOPISTO  
TAMPERE UNIVERSITY OF TECHNOLOGY

SEYEDAMIR AHMADI  
ONTOLOGY VALIDATION OF MANUFACTURING EXECUTION  
SYSTEMS THROUGH THE ANALYSIS OF SEMANTIC DESCRIPTIONS

Master of Science thesis

Examiner: Prof. Jose Luis Martinez  
Lastra  
Examiner and topic approved by the  
Council meeting of the Faculty of  
Engineering Sciences on 28<sup>th</sup> of  
February 2018

## ABSTRACT

### **SEYEDAMIR AHMADI:**

Tampere University of technology

Master of Science Thesis, 67 pages

May 2018

Master's Degree Programme in automation engineering

Major: Factory automation and industrial informatics

Examiner: Professor Jose Luis Martinez Lastra

Supervisor: Borja Ramis Ferrer

**Keywords:** Manufacturing systems, Semantics, Ontology, Knowledge representation, Knowledge-based Systems, Semantic rules, Reasoning, ISA-95, Enterprise-control integration

Current manufacturing systems are comprised of heterogeneous software and hardware components that exchange information on various levels. These levels have distinct functionalities and target different timeframes but they have to communicate for the effective and efficient operation of an enterprise. On one hand, the present trend in industry 4.0 promotes smart manufacturing systems. On the other hand, new product variants, assets, machinery, and diverse manufacturing technologies are constantly added to the manufacturing systems. Hence, the capability of a manufacturing system to follow the dynamic changes of the industry and customers becomes essential. In order to realize this, integration is required to link those individual levels, such as Enterprise Resource Planning (ERP) and Manufacturing Execution Systems (MES), and subsequently perform physical operations in the shop floor. In that sense, using standards becomes significant in order to avoid inconsistent and redundant systems and integration architectures. The ISA-95 standard, from the International Society of Automation (ISA), describes the interface needed for integration of enterprise and control levels by specifying a uniform terminology and a coherent collection of concepts and models.

The objective of this thesis work is to demonstrate an approach for designing a generic manufacturing systems model using a Knowledge Representation and Reasoning (KR&R) formalism, i.e., an ontology, conformant to ISA-95 that allows easy extendibility. The main contribution of the approach lies in the addition of standard and use case specific semantic rules that connect the core concepts and increase the expressivity and reasoning capabilities of the model. Ontologies are flexible and easy to update and enable the reuse of knowledge, which should be considered with the abundance of data available in modern systems. The proposed model describes the system based on products, processes, and resources involved in manufacturing. The applicability, extendibility, and reusability of the proposed model has been validated by its application in an industrial use case as a proof of concept.

## PREFACE

The thesis development and writing process was full of high and low moments that I will always keep fondly with me, because they helped me in becoming a more educated and mature person. The following thesis work would not be possible without the input and support of many individuals, whom I will try my best to thank one by one.

First of all, I am grateful to Professor Jose Luis Martinez Lastra for giving me the opportunity to work in FAST-Lab and providing me with the chance to observe, learn, apply, and grow as part of the FAST team. Additionally, I would like to thank Dr. Andrei Lobov, for guiding me all the way through the master studies, course by course and by bringing prosperity to every course. Anne deserves special thanks for helping me during my research and thesis work in any possible way and more so because FAST-Lab is a better place because of her presence and support.

I would like to give a special mention to Dr. Borja Ramis Ferrer, who's dedication and guidance pushed me step by step through the thesis, and for being understanding and supportive in all situations.

I am very thankful to my dear friend Jacqueline, who has been there for me, in more instances that I can count and knows me more than I like to admit. Arsalan takes a very special place throughout my stay here in Finland and has supported me through thick and thin and is one of the best persons anyone can have the good fortune to meet and have them as friends.

I would also like to thank my friends and office mates Balaji and Wael for being great colleagues and even better friends. I am grateful for having Naveen and Luis as friends, especially along these past few months, providing me with their support and great company. In addition, I am thankful to my friend and flat mate Mohammad, for his invaluable companionship and understanding.

I would like to thank Elena for bringing joy to my life and helping me shape some of the most memorable moments of my life.

Finally, I extend my greatest gratitude to my parents and brother who are the cornerstones of my life. I dedicate this thesis to them, who's endless love and support has given me more in life that I can ever repay.

Tampere, 23.05.2018

Seyedamir Ahmadi

## CONTENTS

1.	INTRODUCTION .....	1
1.1	Problem definition .....	2
1.2	Objectives.....	3
1.3	Limitations and assumptions .....	3
2.	STATE OF THE ART.....	4
2.1	Overview of manufacturing systems .....	4
2.1.1	Enterprise Resource Planning .....	4
2.1.2	Manufacturing Execution Systems .....	5
2.1.3	Industry 4.0 in manufacturing .....	6
2.2	Enterprise-control Integration .....	7
2.2.1	Standards in smart manufacturing .....	7
2.2.2	ISA-95 .....	8
2.2.3	Functional Hierarchy Model.....	9
2.2.4	Role-based Hierarchy Model.....	10
2.2.5	Physical Asset Equipment Model .....	12
2.2.6	Manufacturing activity models .....	13
2.2.7	Object models and attributes .....	14
2.2.8	State of the art of ISA-95 based research.....	16
2.3	Knowledge Representation and Reasoning.....	18
2.3.1	Knowledge management .....	18
2.3.2	Semantics.....	19
2.3.3	Ontology.....	19
2.3.4	Semantic Web.....	20
2.3.5	Web Ontology Language.....	20
2.3.6	Reasoning .....	21
2.3.7	Semantic Web Rule Language.....	21
3.	METHODOLOGY.....	23
3.1	Manufacturing model.....	23
3.1.1	Hierarchy sub-ontology.....	23
3.1.2	Properties of hierarchy sub-ontology .....	24
3.1.3	Operation Type sub-ontology .....	25
3.1.4	Properties of operation type sub-ontology .....	28
3.1.5	Resource sub-ontology .....	28
3.1.6	Properties of resource sub-ontology .....	29
3.2	Ontology editor.....	31
3.3	Remarks of approach .....	31
4.	IMPLEMENTATION .....	32
4.1	Implementation of manufacturing systems model .....	32
4.1.1	The Hierarchy sub-ontology implementation.....	32

4.1.2	The Operation Type sub-ontology implementation.....	33
4.1.3	The Resource sub-ontology implementation.....	33
4.1.4	Enterprise Control Ontology (ECO) .....	34
4.2	Use case implementation .....	35
4.2.1	FASTory line .....	35
4.2.2	General FASTory equipment instantiation.....	37
4.2.3	FASTory zone-transfer instantiation.....	40
4.2.4	FASTory work master instantiation.....	41
4.2.5	Fixed FASTory layout.....	43
4.3	Semantic rules .....	44
4.3.1	Operational SWRL rules .....	45
4.3.2	Product requirement SWRL rules .....	46
5.	RESULTS AND DISCUSSION .....	51
5.1	Operational case .....	51
5.2	Product requirements case .....	54
6.	CONCLUSION AND FUTURE WORK.....	61
6.1	Conclusion.....	61
6.2	Future work .....	62

## LIST OF FIGURES

<i>Figure. 1: Vertical and horizontal integration with MES [14]</i> .....	5
<i>Figure. 2: Collection of integration standards [2]</i> .....	7
<i>Figure. 3: The functional hierarchy model [25]</i> .....	10
<i>Figure. 4: Role-based equipment hierarchy [25]</i> .....	11
<i>Figure. 5: Example of relationship between role-based equipment model and physical asset equipment model [25]</i> .....	13
<i>Figure. 6: Generic activity model of MOM [27]</i> .....	14
<i>Figure. 7: Role-based equipment object model [26]</i> .....	15
<i>Figure. 8: Knowledge Engineering [41]</i> .....	19
<i>Figure. 9: Hierarchy sub-ontology model</i> .....	24
<i>Figure. 10: Activity model of production operation management [27]</i> .....	26
<i>Figure. 11: Operation Type sub-ontology model</i> .....	27
<i>Figure. 12: Resource sub-ontology model</i> .....	29
<i>Figure. 13: Hierarchy sub-ontology implementation</i> .....	32
<i>Figure. 14: Operation Type sub-ontology implementation</i> .....	33
<i>Figure. 15: Resource sub-ontology implementation</i> .....	33
<i>Figure. 16: Sub-ontology imports</i> .....	34
<i>Figure. 17: Enterprise Control Ontology (ECO) implementation</i> .....	35
<i>Figure. 18: FASTory assembly line</i> .....	36
<i>Figure. 19: a) Keyboard patterns b) Frame patterns c) Screen patterns [61]</i> .....	36
<i>Figure. 20: FASTory simulator [60]</i> .....	37
<i>Figure. 21: Extensions of ECO model</i> .....	39
<i>Figure. 22: Order1 assembly requests</i> .....	40
<i>Figure. 23: a) Work zones in WC7 b) Work zones in WC1 c) Work zones in WC2- 6 / 8-12 [61]</i> .....	40
<i>Figure. 24: Assembly zone transfer 5</i> .....	42
<i>Figure. 25: Bypass zone transfer 5</i> .....	42
<i>Figure. 26: Loading zone transfer in WC1</i> .....	43
<i>Figure. 27: Modified functionality of robot 4</i> .....	44
<i>Figure. 28: FASTory instance inferences</i> .....	51
<i>Figure. 29: Robot1 instance inference</i> .....	52
<i>Figure. 30: Job order inferences</i> .....	52
<i>Figure. 31: Work directive inference</i> .....	53
<i>Figure. 32: Process segment inference</i> .....	53
<i>Figure. 33: Order1 product requirements</i> .....	54
<i>Figure. 34: Query of bypass zone-transfers of Order1</i> .....	55
<i>Figure. 35: Order2 product requirements</i> .....	56
<i>Figure. 36: Query of assembly zone-transfers of Order2</i> .....	56
<i>Figure. 37: Order3 product requirements</i> .....	57
<i>Figure. 38: Query of assembly zone-transfers of Order3</i> .....	58

*Figure. 39: Order4 product requirements* ..... 59

*Figure. 40: Order5 product requirements* ..... 60

## LIST OF TABLES

<i>Table. 1: Attributes of equipment [26]</i> .....	15
<i>Table. 2: Attributes of equipment property [26]</i> .....	16
<i>Table. 3: Properties of hierarchy sub-ontology</i> .....	25
<i>Table. 4: Properties of operation type sub-ontology</i> .....	28
<i>Table. 5: Properties of resource sub-ontology</i> .....	30
<i>Table. 6: Work cell 5 zone-transfers</i> .....	41
<i>Table. 7: Updated FASTory configuration</i> .....	44
<i>Table. 8: Operational SWRL rules notation</i> .....	45
<i>Table. 9: Product requirement SWRL rules</i> .....	47



## LIST OF SYMBOLS AND ABBREVIATIONS

BPMN	Business Process and Model Notation
ERP	Enterprise Resource Planning
ISA	International Society of Automation
ISO	International Organization for Standardization
KB	Knowledge base
KBS	Knowledge-based Systems
KPI	Key Performance Indicator
KR&R	Knowledge Representation and Reasoning
MAS	Multi Agent System
MES	Manufacturing Execution Systems
MI	Manufacturing Intelligent
MOM	Manufacturing Operations Management
OPC UA	OPC Unified Architecture
OWL	Web Ontology Language
PCS	Process Control Systems
RDF	Resource Description Framework
RDF-S	Resource Description Framework Schema
REA	Resource Event Agent
SCADA	Supervisory Control and Data Acquisition
SOA	Service Oriented Architecture
SW	Semantic Web
SWRL	Semantic Web Rule Language
TUT	Tampere University of Technology
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	Extensible Markup Language

# 1. INTRODUCTION

Modern manufacturing systems are multi-layered systems that depend heavily on the level of integration, interoperability, and compatibility of their individual sub systems. Two distinctive areas in such systems are the enterprise and business level, and the manufacturing operations level that target different set of objectives and time horizons. In a production systems hierarchy, Manufacturing Execution Systems (MES) link the Enterprise Resource Planning (ERP) with the lower levels of the shop floor, e.g., Supervisory Control and Data Acquisition (SCADA) and subordinate controllers and machines. This connection provides runtime data about the events that happen in the shop floor to managers, workers, and related functions in charge of information exchange. An effective and complete support of the integration between the above-mentioned systems should cover each of their underlying elements, both in terms of software and hardware. Additionally, it is necessary to identify the interactions between these elements, which could directly relate to the domain to attest the systems full range of operation and to the users, to check that the proper status is communicated through each components interface. Some of the benefits of functional and non-functional validation of such an integration are early detection of errors, increased insight into process and system performance [1].

Vertical and horizontal integration, i.e., integration between the higher and lower levels of an enterprise (intra-level) and integration amongst the elements within a level (inter-level), is necessary for the effective operation of manufacturing systems. On this account, the application of appropriate interfaces and infrastructures linking those functional groups becomes important to facilitate, proper information exchange. Taking into consideration the higher-level complexity of modern manufacturing systems, standards have emerged as well-established solutions, promoting uniform set of guidelines, rules, and definitions. Some of the advantages of conforming to standards include the removal of technological difficulties, introducing new market opportunities, and economic growth [2].

System architects or engineers responsible for modelling different functional hierarchies and information flows of a system, have to take into account an abundance of data. Hence, with the existence of such amounts of data in modern systems, available from heterogeneous components, the Semantic Web (SW) and its affiliated technologies act as an effective enabler for data organization and its use.

According to World Wide Web Consortium (W3C), the SW offers a standard framework for the distribution and reuse of data throughout different applications, businesses,

and community boundaries [3]. Semantic technologies are the tools used for storing and manipulating these data. The higher interoperability of semantic web technologies within these modern systems serves the purpose of adaptability and reusability of components. In this context, engineers can use Knowledge Representation and Reasoning (KR&R) formalisms such as ontologies, as a means of modelling (structuring) the knowledge with formal definitions, taxonomies, and relationships in any domain [4]. The structured knowledge could then be shared across other elements of the system. Therefore, validating the knowledge and its modelling is necessary to ensure the proper operation of such Knowledge-Based Systems (KBS) [5][6].

## 1.1 Problem definition

Reconfiguration of manufacturing systems due to the addition of new technologies or equipment is a common area of focus that has been present for a long time. This task becomes harder when there are inconsistencies in the basic definitions of elements and their communications. In the domain of KR&R, ontologies represent a good solution for modelling the system and tackling the issue of re-configurability at software level.

Ontologies provide the means for structuring the knowledge available in different domains and are used in KBS. To accurately validate the modeling, complete performance, and operational range of a system, one needs proper understanding of the knowledge model and its functionality. In order to do so, the core elements of the system and the connection amongst them (i.e., the structure of the knowledge), and the functionalities they perform, should be identified [7]. It is in this stage, where clear understanding of the domain of discourse becomes significant to avoid multiple and redundant architecture proposals [8]. This in turn helps with the precise identification and representation of knowledge axioms, i.e., the classes, relationships, and entities that shape the knowledge structure. In the context of current industrial trends, manufacturing systems have to be flexible to adapt to the dynamic changes that are continuously affecting the industry [9]. Such dynamic changes represent new knowledge in the industrial domain. On the account of KBS, the new knowledge should be added to the Knowledge Base (KB) using established formal modeling methods. Some of the distinct elements present in the domain of manufacturing systems could include but not be limited to the equipment being used, product definitions, resources and their capabilities, personnel, manufacturing activities, etc. The approach presented for modeling a manufacturing systems ontology comprised of these elements should address the following questions:

- How to define concepts, taxonomies, and relationships in a manufacturing systems model, that are uniform and well-established in its domain, in order to allow extendibility and reusability of the model?
- Which properties should be checked within the manufacturing systems model to evaluate its applicability?

- How to enhance the semantic descriptions within the manufacturing systems model?
- How and in which aspects does the use of semantic rules increase the reasoning capabilities of the manufacturing model?

## 1.2 Objectives

The work presented in this thesis work aims to achieve the following objectives:

- To identify standard conformant generic concepts, taxonomies, and relationships for describing the core elements in the domain of manufacturing systems.
- To model the manufacturing system based on:
  - The main elements of manufacturing system that have business and manufacturing (operational) value.
  - The different functional levels of the system hierarchy, covering the interface of enterprise-control integration.
  - Taking into account a modular modelling approach, allowing easier modifications of the underlying segments without referring to the more complex and consolidated manufacturing model.
- To add supporting standard and use case specific semantic rules to the manufacturing model to enable the:
  - Connection of related concepts and modules.
  - The achievement of specific system goals.
- To demonstrate the proposed manufacturing ontology in the industrial use case, highlighting the solutions applicability and extendibility within the use case presented, in addition to the presentation of the use case specific semantic rules that ultimately enable the identification of the necessary resources and operations required for performing production operations.

## 1.3 Limitations and assumptions

The modelling of the proposed manufacturing system is based on the guidelines and concepts present in the referenced standard. The purpose of this work is not to fully cover every aspect of it, but to create a generic model based on the most dominant functions of the standard, and to add and extend some areas with use case specific implementations. The research done in this work is in the domain of factory automation. The manufacturing systems modeling and the addition of semantic rules is performed based on the systems components and dependencies. The industrial use case presented in this research is used as a test basis to demonstrate a proof of concept for the proposed solution, hence it is assumed that the users of the model have an understanding of the system, its input, and description within the factory automation domain.

## 2. STATE OF THE ART

This chapter describes a literature review on some of the technologies, models, and industrial applications related to manufacturing systems, enterprise-control integration, and knowledge representation and reasoning.

### 2.1 Overview of manufacturing systems

Modern manufacturing systems as a whole are composed of characteristic elements, features, and levels that help in identifying the system. These characteristics are not just related to the general definitions of the production type and products. On one hand, manufacturing can be expressed as the total of products, process, and resources [10]. Hence, having control over their activities and interactions becomes necessary. On the other hand, the system has to be defined based on various levels of the hierarchy, from the enterprise level (e.g., ERP) to the control layer (e.g., MES) and subsequently with the shop floor, where the physical equipment is located. A common terminology and language becomes very important to streamline the structural hierarchy of these systems while enabling better communication between all parties.

#### 2.1.1 Enterprise Resource Planning

The availability of the right information, at the correct and required time, is essential for management of business processes. Furthermore, the information flow and accuracy of it affects the decision-making capabilities of an enterprise, helping them to understand the operation process, and prevent loss of profit. ERP systems are centralized systems, which considerably enhance the organizational management by aggregating all essential business processes and data flows and facilitating the flow of information within all divisions of the enterprise.

Besides streamlining workflows between various divisions and decreasing the costs related with duplication of information within systems, the utilization of ERP systems affect the following aspects [11] [12]:

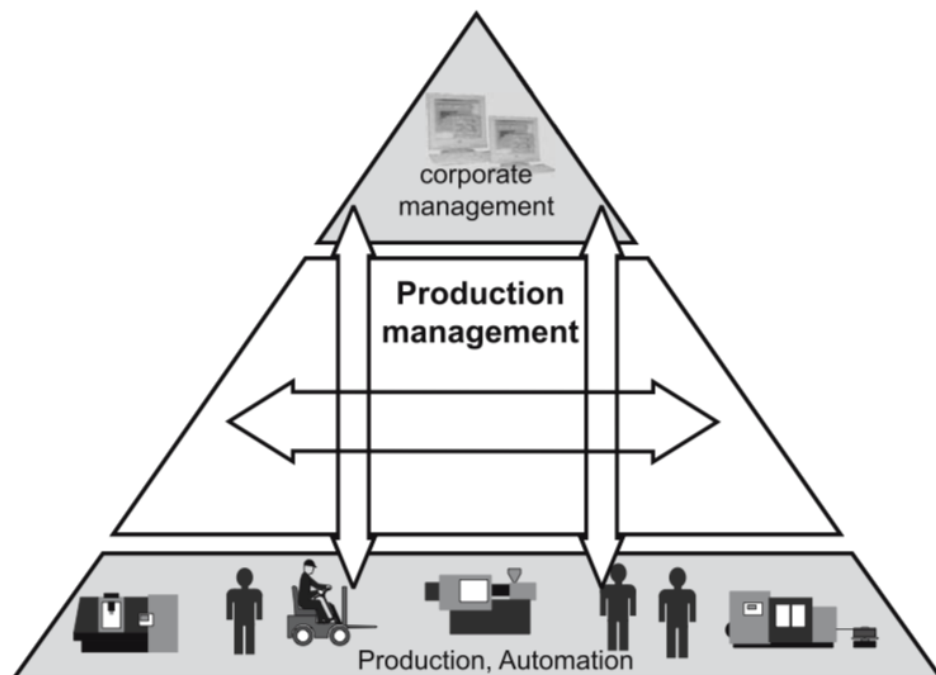
- Operational: Automating the business processes, hence increasing productivity
- Managerial: Improving data analysis abilities
- Strategic: Supporting the business growth
- IT systems: Implementing standard operation methods in all business units, adding value to business flexibility

- Organizational aspect: Learning about business aspects, inspiring users, and building a collective vision

### 2.1.2 Manufacturing Execution Systems

Manufacturing Execution Systems (MES), can be defined as systems that manage the control of processes by interfacing the higher decision-making levels of a system (e.g., ERP) with the physical lower levels of an enterprise. In doing so, MES has the responsibility of comprehensive scheduling of tasks in a manufacturing system, from initiating orders, responding to various events, modification of plans, and to follow up on tasks [13]. MES supports the effective functioning of a company with vertical and horizontal integration. By positioning MES between the corporate management (ERP) and the lower level production, vertical integration is achieved. In the three-level based hierarchy shown in Figure. 1, the production level receives up to date information from ERP, while production information is sent back to ERP, both through the MES level.

Within the domain of production management, MES helps coordinating between production, personnel, and quality, the three main functional groups in this layer. IT solutions, help to map these functional groups, for them to use a single data pool and hence, perform in an achievable uniform manner. The mappings should prevent obtaining redundant and duplicated information and transactions. These all translate into the so called horizontal integration [14].



*Figure. 1: Vertical and horizontal integration with MES [14]*

The scale of an enterprise or the specific industry that the MES is being implemented in is less important than the production structure (e.g., production segment or assembly line, etc.) of enterprise. Because of its modular organization, MES can be simply implemented to a particular production setting and its required tasks. In order to do so, i) the initial production structure needs to be identified and ii) identification of how the current production scheduling and control is being performed and what extensions the MES functionalities provide, is important as well.

### **2.1.3 Industry 4.0 in manufacturing**

The Fourth Industrial Revolution, better known as industry 4.0 [15], is heavily shaping the future industrial developments of manufacturing. Smart factories, intelligent manufacturing processes, and cyber physical systems (CPS), are some of the main objectives of what industry 4.0 is trying to accomplish. Modern manufacturing systems adhering to these changes need to be more flexible and responsive to address the dynamic changes of the industry and the customers. Building a more information-rich and digital infrastructure paves the main path to that flexibility. In order to implement these changes on one hand, industry 4.0 has some planning objectives such as building a reference architecture, efficient management, and improving efficiency of resource usage [16].

On the other hand, the initiative of industry 4.0, recommends that these objectives have to be implemented while realizing three main integrations as follows [17]:

- Vertical integration: Connecting ERP with MES and subsequently with the shop floor [18]
- Horizontal integration: Integration across value networks
- End-to-end integration: Digital integration throughout the value chain

Identifying the main elements of a hierarchy in each level, their functionality, and the intra-inter layer communication exchange presents some of the core challenges while implementing integration. It should be noted that based on the application, these integrations may coincide and be concurrently applied, particularly when they enable communication between autonomous objects and human-machine interfaces [19].

In order to realize and fulfil the above-mentioned objectives, a vast amount of research has been performed, each tackling general or case-specific issues that need to be addressed for proper conformance and implementation of industry 4.0 guidelines. Interoperability of modular and heterogeneous components with established legacy devices, is one of such issues. Choosing the right data exchange formats and standards that could cover elements in diverse automation applications is also of importance. In most cases, more than one of such technologies or standards may need to be used within the same implementation [20]. Interoperability is also relevant in distributed systems. Information Technology (IT) driven paradigms such as Multi Agent Systems (MAS) have emerged

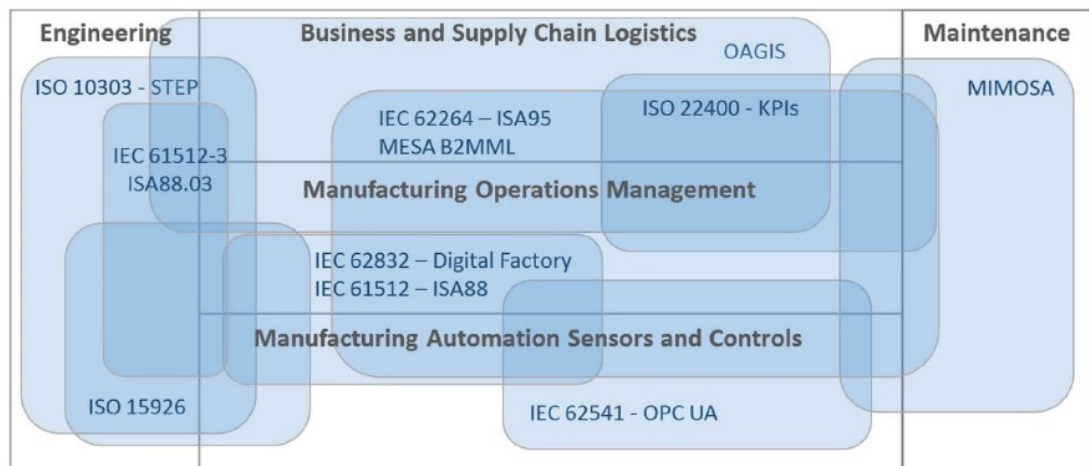
for distributing control and enhancing re-configurability. For doing so, KB can be used for implementing an agent based manufacturing system, promoting agent cooperation and interaction. This is useful for the autonomy of agents, dynamic re-configuration and scheduling of production, supporting flexibility, and agility for modern smart manufacturing in the context of industry 4.0 [21] [22]. Agility might also be related to on demand system modification based on customer feedback and requirements [9].

## 2.2 Enterprise-control Integration

### 2.2.1 Standards in smart manufacturing

As previously discussed in section 2.1.3, the current industrial trend is promoting smart manufacturing methods that need to be addressed in different scopes. Both manufacturers and vendors need to conform to a unified set of concepts, rules, and guidelines in order to produce hardware and software that perform adequately in the face of more complex activities and data flows. This is where standards become essential in smart manufacturing [2]. Some of the benefits of using standards are reduced cost of procurement, flexible organizational structure of human, physical, digital, and information resources in support of the dynamically changing business requirements, and universal access to trustworthy data through a products life cycle.

Horizontal, vertical, and end-to-end integration are key challenges in the current scope of manufacturing [17]. Figure. 2, shows a collection of some of the prominent international standards available in the domain of manufacturing and furthermore illustrates which kind of integration they target.



**Figure. 2:** Collection of integration standards [2]

System architects and engineers have the freedom to choose the standards specific to their industrial application and to extend them or as in some cases, discussed later as



part of state of the art of ISA-95-based research, use other standards to complement their solution.

### 2.2.2 ISA-95

The International Society of Automation (ISA), is a professional association specializing in automation and control systems and develops international standards in this domain, besides training, educating, and certifying professionals [23]. ISA started developing ISA-95 in the 1990's, when the gap between ERP systems and Process Control Systems (PCS) layer was a topic of concern and bridging this gap was necessary for communication between systems and people. Hence, the standard was developed to facilitate the integration of enterprise and control systems for decreasing cost, risk, and errors associated with the mentioned integration [24].

This International Organization of Standardization (ISO) has adopted the same standard under the name IEC-62264. For the use of this thesis work, ISA-95 will be used for reference where ever needed. The standard as its currently envisioned consists of seven parts as follows:

**ISA-95.00.01-2010 – Enterprise-Control System Integration – Part 1** (Models and Terminologies): The first part describes the interface between enterprise and manufacturing functions [25].

**ISA-95.00.02-2010 – Enterprise-Control System Integration – Part 2** (Object Model Attributes): The second part describes the attributes and the object models for the information exchange of the elements of part one [26].

**ISA-95.00.03-2013 – Enterprise-Control System Integration – Part 3** (Activity models of Manufacturing Operations Management): The third part defines, predominant activity models available in the scope of operations management [27].

**ISA-95.00.04-2012 – Enterprise-Control System Integration – Part 4** (Objects and Attributes for manufacturing operations management): The fourth part defines attributes and object models for the elements of manufacturing operation management defined in part three [28].

**ISA-95.00.05-2013 – Enterprise-Control System Integration – Part 5** (Business-to-Manufacturing Transactions): The fifth part defines the information exchange of the functions performing manufacturing and business activities amongst level three and four, and inside level three [29].

**ISA-95.00.06-2014 – Enterprise-Control System Integration – Part 6** (Messaging Service model): The sixth part defines a set of services that could be used for exchange-

ing information messages for the interface of manufacturing and business activities [30].

**ISA-95.00.07-2017 – Enterprise-Control System Integration – Part 7** (Alias Service model): The seventh part defines an alias service model for mapping elements of communicating applications [31].

In the scope of this thesis, parts one, two, three, and four are the main areas of interest and the following sections each define further elements of each part. Additionally, it should be taken into account that the terms “ISA-95” and “Standard” will be used interchangeably in this thesis work. As explained above, part one of ISA-95, defines the interface between enterprise and manufacturing and control activities. In other words, vertical integration between the ERP and MES levels is the main area of focus. In order to do so, the standard has different hierarchy models, such as functional hierarchy, role-based hierarchy, and a physical asset equipment hierarchy, each identifying the elements of the interface based on different aspects. These models are beneficial for those involved in design, creation, and integration of automation products aimed for the interface of enterprise and control layers, because of the general and abstract views of the system that they provide.

### 2.2.3 Functional Hierarchy Model

The functional hierarchy model of the standard, demonstrates the different levels of a system based on their specific functionality, and within their corresponding timeframes as shown in Figure. 3. The scope of part one is the interface between level four and level three of the hierarchy (i.e., vertical integration). Levels zero, one, and two, commonly referred to as the shop floor, depict how and where the actual production, sensing and handling of processes, and the control functions in charge of decision making are distributed. These levels are common to all types of batch, continuous, and discrete operation types. Detailed functionalities of every element of the aforementioned levels (particularly the lower levels) are beyond the scope of the standard and this implementation. But it should be noted that as discussed in section 2.2.1, ISA-95 is used in conjunction with other standards (e.g., ISA-88) to supplement implementation scenarios.

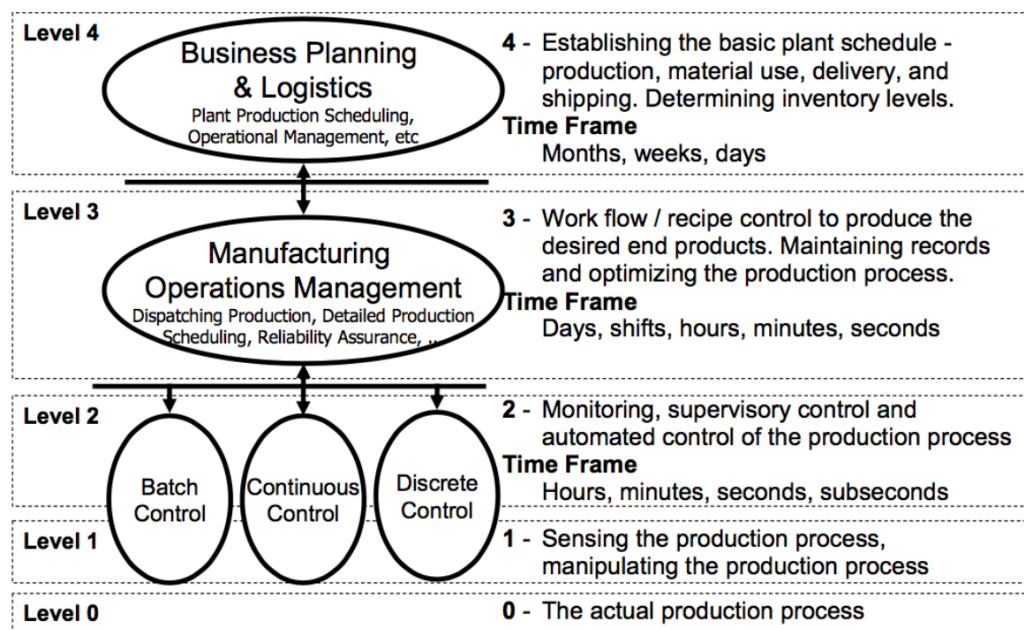
The third level, known as Manufacturing Operation Management (MOM), which corresponds to functionality of an MES, may include some of the following activities amongst others:

- Controlling manufacturing operations
- Controlling material storage and movement
- Gathering and maintaining production, inventory, resource, quality, and energy use area data

- Transforming the “business oriented” level 3-4 information into more “manufacturing operations oriented” information for use within MOM

The fourth level, known as business planning and logistics (corresponding to ERP), may include some of activities as follows:

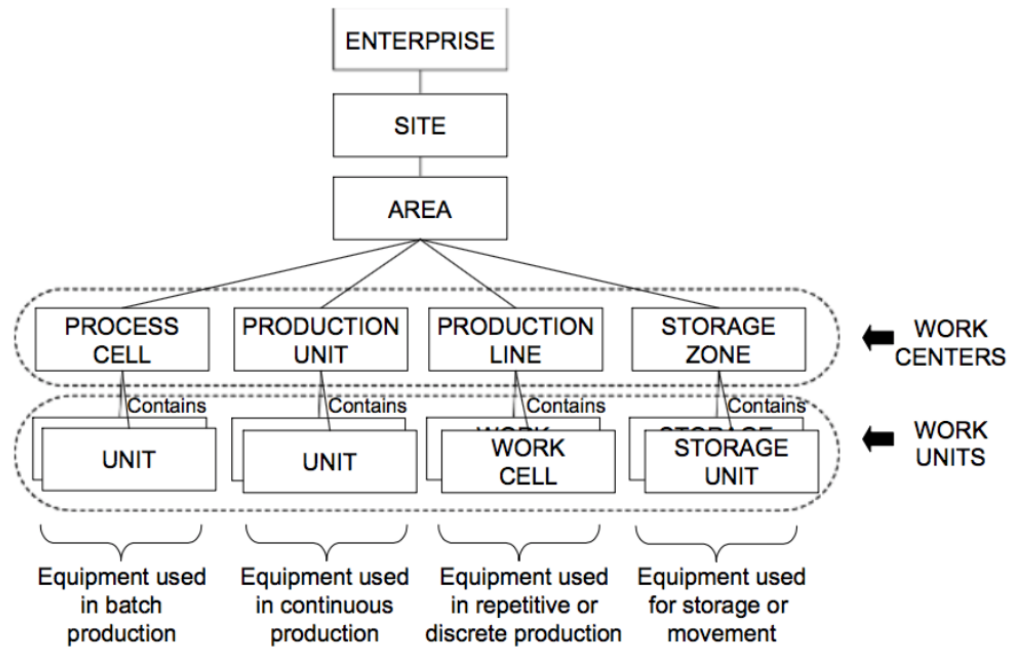
- Creating basic plant production schedule (material usage, the delivery, and shipping)
- Gathering and maintaining equipment use and history data for planning preventive and predictive maintenance
- Modifying production schedule based on inputs from other levels



*Figure. 3: The functional hierarchy model [25]*

## 2.2.4 Role-based Hierarchy Model

ISA-95 distinguishes the main assets of the enterprise involved in manufacturing, based on their functionality and the specific activities they perform, or their physical configuration, location, and maybe relationship with other resources. The former is identified in the role-based equipment hierarchy as shown in Figure. 4.



**Figure 4:** Role-based equipment hierarchy [25]

The main areas of responsibility in a role-based hierarchy are identified as follows:

**Enterprise:** The highest level of the hierarchy, which typically involves identifying the products that will be manufactured, and how and in which sites the manufacturing operations take place. Enterprise is comprised of sites and areas and involves level four functions.

**Site:** Sites are based on physical, geographical, or categorizations done by the enterprise. Sites maybe comprised of areas and several other sub sections of area, as discussed in the next groupings. The location and core production capabilities of the site, help with identifying a site. Local site management and optimization are where the level four functions are involved.

**Area:** Similar to sites, areas are also based on their physical, categorizations, and other site-specific classification. Level three functions are generally performed in an area. Areas are comprised of work centers and work units, representing the lower sub-levels of an area, and have distinct manufacturing abilities and capacities.

**Work Center:** Work centers are equipment elements categorized under an area. ISA-95 defines particular terminologies for work centers and work units within the domain of MOM that are applied to discrete manufacturing, batch, continuous production, storage, and equipment/material movement. The main types of work centers outlined in the standard are, process cells (for batch production), production units (for continuous production), production line (for discrete production), and storage zones (used for materi-

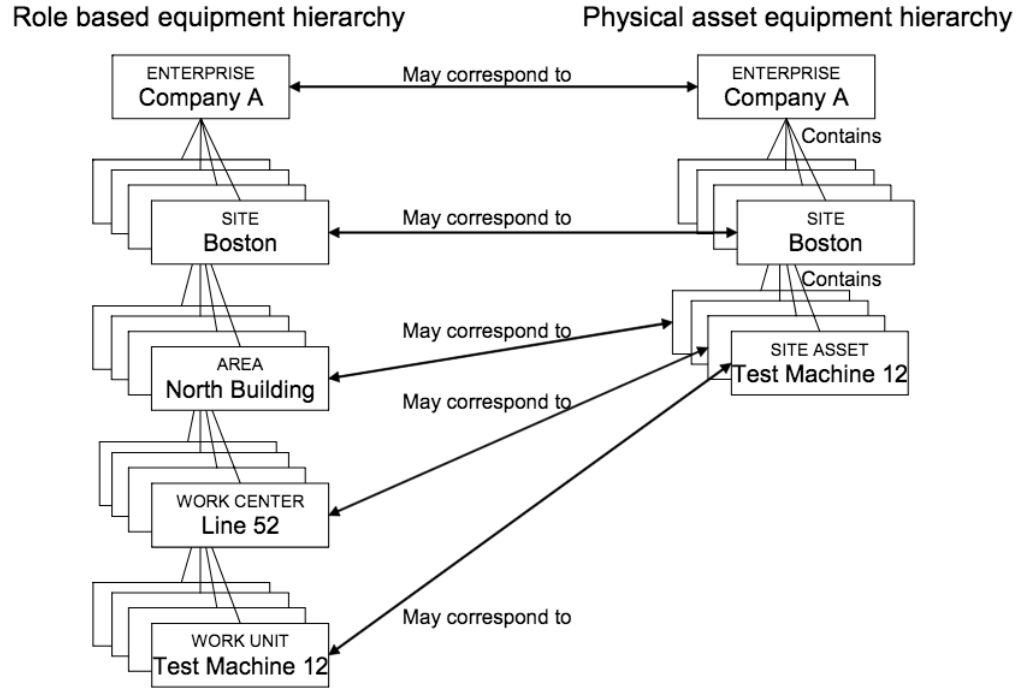
al/equipment storage and movement). Work centers may be made up of one to several types of work units.

**Work Unit:** Work units are the lowest level of equipment in the role-based equipment hierarchy, defined under a work center and usually scheduled by level three functions. The main types of work unit in ISA-95 are defined as unit (for batch and continuous production), work cell (for discrete production), and storage unit (for material/equipment storage and movement).

The areas of responsibility in the role-based hierarchy demonstrate the types more relevant to the enterprise and business planning (level four), and manufacturing operations management (level three). Although the elements of the above-mentioned categorization have well-established boundaries, each one may deal with activities of the other level and vice versa when needed. For example, while level four functions usually deal with the enterprise and site definitions, some of their functions may cross into the activities within an area. On the other hand, while level three functions, are typically performed within an area, and their subordinate work centers and work units, they may coordinate with level four functions for performing their operations.

### 2.2.5 Physical Asset Equipment Model

The assets of an enterprise associated with the manufacturing, characterized by their configuration, location, and relationship with other physical assets are identified in the physical asset equipment model. This identification may also be related to financial aspects of interest to the enterprise. As previously discussed, although the role-based equipment model and the physical asset equipment model describe the assets of enterprise from two different aspects, these models can overlap at any level as shown in Figure. 5. The physical asset model usually includes more levels corresponding to a physical assembly or cost center hierarchy and may have different names as well, such as Site Asset in the figure below.



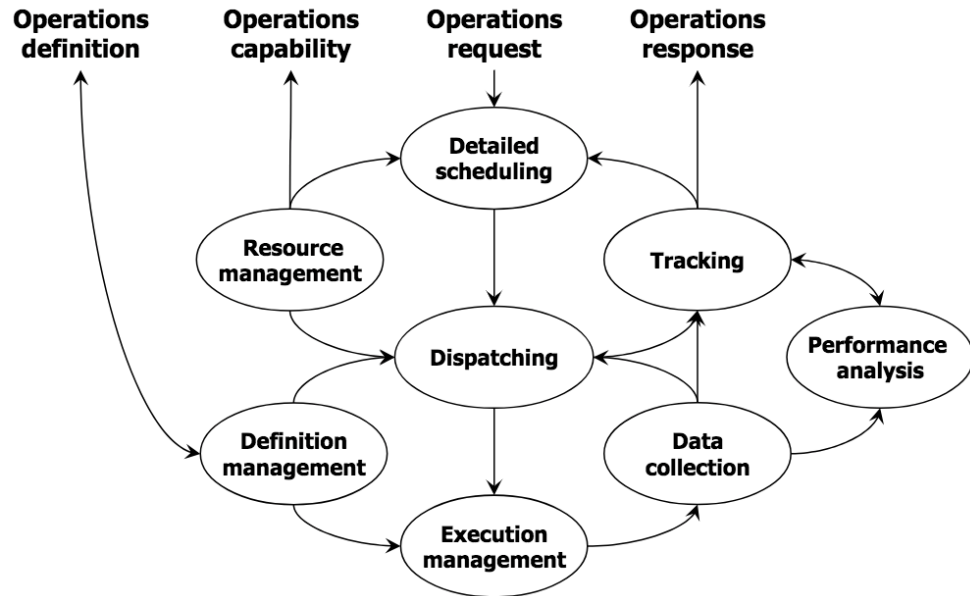
**Figure. 5:** Example of relationship between role-based equipment model and physical asset equipment model [25]

## 2.2.6 Manufacturing activity models

The third part of ISA-95, defines activities within the manufacturing operations management (MOM) level, also known as level three of the functional hierarchy model. MOM coordinates the tasks of the work force, material, equipment, and energy for transforming unprocessed material and segments into products. These activities could be executed by the equipment, work force, and information systems and in doing so exchange information with level four and level two activities. The main types of MOM activities are defined as production, maintenance, quality, and inventory operations management. The standard states that other supporting management activities occurring in manufacturing operations might exist such as management of security, configurations, and documents, amongst others. Such supporting activities are enterprise specific and are not further explained in ISA-95.

For each type of MOM activity, ISA-95 defines a generic set of activity models (as a collection of tasks) that exchange information in various stages of the production as shown in Figure. 6. These stages correspond to a request-response cycle that starts with the request (scheduling), transforming the request into work schedules, dispatching the work based on the schedule, followed by work execution management, recording data, and finally converting and sending the recorded data back as responses. The state and activities of each of these categories can be updated based on current feedback from within level three. For example, the data tracking activity could update the scheduling

activity based on current resource capacities available and the actual state of production and etc.



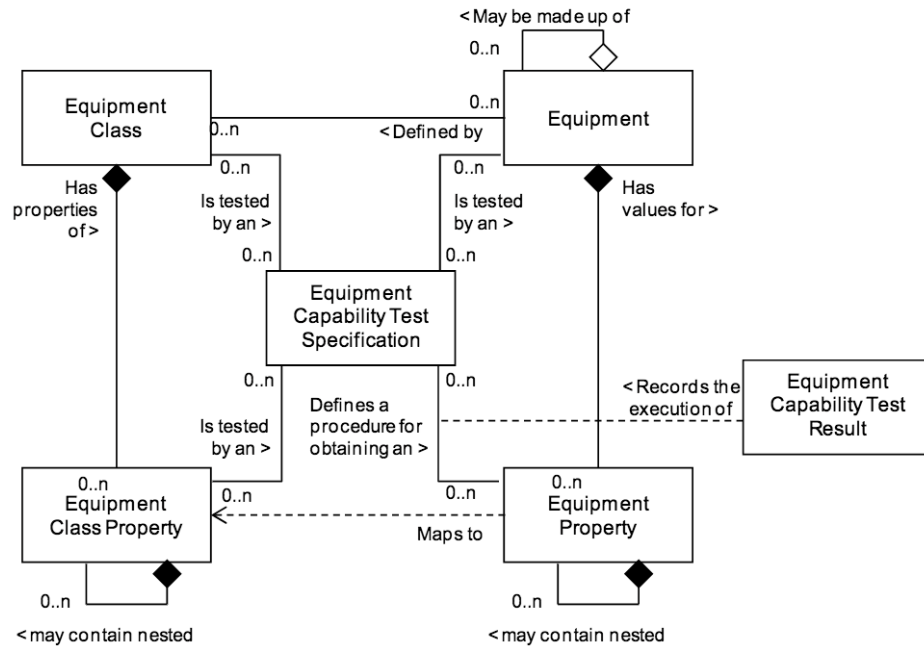
*Figure. 6: Generic activity model of MOM [27]*

In the context of the current work, the production operation management activity model is the main area of focus and is further discussed in chapter 3.

### 2.2.7 Object models and attributes

ISA-95 uses the Unified Modelling Language (UML) for representing the object models and attributes of elements of the integration interface and work models. As previously discussed, part two of ISA-95 defines the object models and attributes for the main elements described in the interface of enterprise-control integration (part one), while part four defines the object models and attributes for the activity models in MOM (part three). ISA-95 defines a minimum collection of industry independent characteristics, which might be used according to the actual implementation. In cases where more application specific information is required, such information can be added as object properties. The standard uses this practice to allow the use of standard attributes, which can be complemented by the supporting object properties that enhance flexibility and extensibility.

For example, Figure. 7, shows the object model of role-based equipment (from part one), providing information about particular equipment classes, equipment instances, and equipment capability tests. It should be taken into consideration that the word “class” is not meant in the sense of UML classes but it’s used to represent a category.



**Figure. 7:** Role-based equipment object model [26]

On this account, equipment class demonstrates the grouping of equipment that share similar features for any purpose, such as “reactor unit”, for instance. Equipment itself on the other hand, demonstrates an element of the object model and could be an area, production unit, storage unit, and etc.

Table. 1, shows the list of attributes for specific equipment with some defined examples:

**Table. 1:** Attributes of equipment [26]

Attribute Name	Description	Production Examples	Maintenance Examples	Quality Examples	Inventory Examples
<b>ID</b>	A distinct identification of a particular piece of equipment, within the extent of exchanged information	Jig 347	Wldr445	SN3883A T	VIN28203
<b>Description</b>	Additional information about the equipment	This is the east side, north building, widget jig	Welder for north building	Floor 2 lab auto titrator	Shipping dock lift truck
<b>Equipment Level</b>	An identification of the level in the role-based equipment hierarchy	Production line	Work Center	Site	Area



Table. 2, shows the list of extended attributes, also called properties, for specific equipment with some defined examples.

**Table. 2:** *Attributes of equipment property [26]*

<b>Attribute Name</b>	<b>Description</b>	<b>Production Examples</b>	<b>Maintenance Examples</b>	<b>Quality Examples</b>	<b>Inventory Examples</b>
<b>ID</b>	An identification of the specific property.	Run Rate	Capacity	Resolution	Max Weight
<b>Description</b>	Additional information about the Equipment property	Widget making average run rate	Capacity of the welder	Minimum peak resolution	Maximum carrying weight for the truck
<b>Value</b>	Literal value, list of values, or range of the property	59	{10-200}	0.05	1
<b>Value Unit of Measure</b>	The measuring unit of related value, if applicable.	Widgets/Hour	Amperes	%	Tons

## 2.2.8 State of the art of ISA-95 based research

The different hierarchies and models of ISA-95 provide a solid foundation to system engineers to address the issue of integration throughout their implementation. Integration, could correspond to vertical and horizontal integration, either individually or concurrently, which should consider the identification of core functions and data flows between them. This is further shown in the state of the art of ISA-95 based research performed in the domain. On this account, ISA-95 itself has also been used individually or in combination with other standards and well-established technologies to help resolve research initiatives.

In case of vertical integration for example, the authors of [29], demonstrate the alignment of ISA-95 with IEC-62714, with the goal of proper mapping of linked concepts. IEC-62714 is a standard for data exchange in automation engineering domain, enabling the modelling of the shop floor, which is essential considering that ISA-95 mainly focuses on enterprise and control integration. Hence, by mapping object models of ISA-95 with the CAEX model elements, integration of the MES (level three) and the shop floor (levels zero, one, and two of the functional hierarchy model) becomes possible. The entwined information on the other hand, will give applications accessing this information a more comprehensive outlook on the production model. Horizontal and vertical

integration has been addressed in [33], where a Resource Event Agent (REA) ontology (derived from IEC-15944-4) has been used for modelling the internal activities within the ERP layer, while ISA-95 addresses the connection of ERP and MES levels, representing horizontal and vertical integration, respectively. The alignment of business and production services is one of the outcomes of this concurrent integration. With respect to vertical integration, the proposed solution of [34] is an ISA-95 based Manufacturing Intelligent (MI) system, positioned as an intermediary layer between ERP and MES in the functional hierarchy model, to support lean manufacturing. The standard was used to define the internal architecture of the MI system. By demonstrating the implementation as part of two use cases, the authors assert enhanced responsiveness by delivering real-time view on the operations performed in the shop floor. This in turn has helped with dynamic generation of Key Performance Indicators (KPI), as main contribution of the research work.

Research has also been performed in the MOM domain of ISA-95, which details the activity models involved in different operations such as production, as previously discussed. The authors of [35], use the functional hierarchy and manufacturing operations model of ISA-95 as a basis for describing a methodology for designing Business Process and Model Notation (BPMN<sup>1</sup>) process models, which in turn identifies core activities and data exchange sequences for the integration between ERP and MES systems in the use case presented. As previously stated, ISA-95 can be used with other standards for extending or supplementing the proposed implementation. In case of [36], the design of a batch control management application has been proposed with the use of OPC Unified Architecture (OPC UA<sup>2</sup>) and Service Oriented Architecture (SOA) [37]. The design process involved the production operation management model of MOM in conjunction with ISA-88, which addresses the design and implementation of batch control systems. The authors claim that the data and activity models of ISA-95/88 simplified the design process of the application, facilitating communication between MES and Process Control Systems (PCS).

The research performed on the industrial applications of ISA-95 shows the standards flexibility for its application in modern and smart manufacturing systems. The level of abstraction provided by the standard enables system engineers to adapt any part of the standard applicable to their specific use case without restrictions. This is shown in the examples above, where the researches have chosen to utilize the standard to either address vertical or horizontal integration alongside other standards, or to map object models and attributes of one standard with another to supplement them. It should be noted that depending on the level of application specific implementation and extensions, in-

---

<sup>1</sup> <http://www.bpmn.org/>

<sup>2</sup> <https://opcfoundation.org/>

consistencies or loss of conformance to the standards could be a probable outcome that needs to be addressed and taken into consideration.

## **2.3 Knowledge Representation and Reasoning**

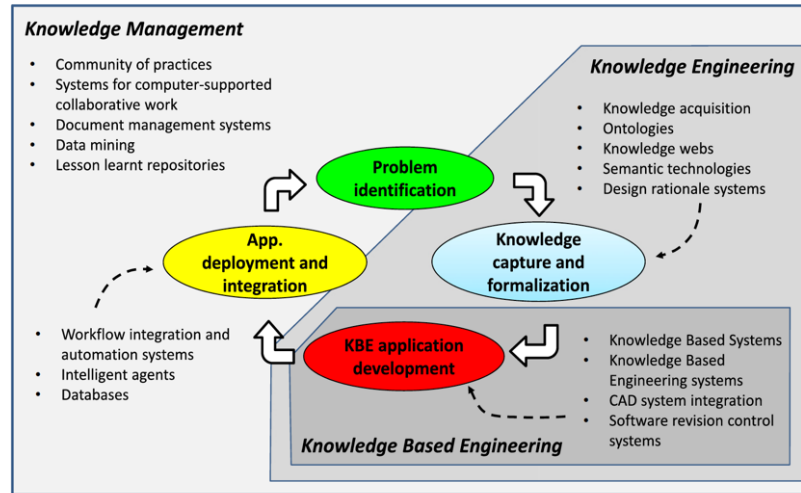
Knowledge Representation and Reasoning (KR&R) is applied in different fields such as medicine, economics, and industrial automation. In the context of modern manufacturing systems and the current trends of the domain influenced by industry 4.0, KR&R represents a viable solution for automatic control and monitoring of systems. The following section aims to describe some of fundamental concepts in the field of KR&R.

### **2.3.1 Knowledge management**

Knowledge management is an essential aspect in every system and allows the description of domain knowledge. This is significant in modern manufacturing systems for instance because of the heterogeneous nature of different components that exchange information on various levels of the systems functional hierarchy. Knowledge management is defined as explicit and implicit knowledge (core strategic resources of an organization), with the goal of improving the handling of knowledge with the use of other fields such as Information Technology (IT) and business process management [38]. For facilitating this management, a KB is required to properly structure the underlying knowledge and a set of rules complementing it to enable inference engines to reason over the KB.

Knowledge based Systems (KBS) are systems that use a KB for the storage of information and querying it when needed, and to find solutions to complex problems in specific domains. Experts from such domains are required to facilitate the storage of knowledge and that is why, KBS were originally also referred to as “expert systems”. Another main part of such a system is an inference engine, which is used in conjunction with the KB. This engine accesses the KB and performs reasoning mechanism based on its underlying rules. For proper utilization of the engine, structuring the data in KB becomes important. Originating in the 1970s, one of the earliest examples of such an application, called “MYCIN” [39], was used as a diagnosis tool that suggested treatments for blood infections based on all the data in the KB provided by a physician. The KB was implemented as a set of IF-THEN rules, which in turn meant that certain actions would be suggested by the system if a specific criteria or combination of criteria was met [7] [40].

The main stages for developing a KBS are explained as the identification of the problem, an acquisition method, and a structuring (codification) method before being able to proceed with the integrated solution [41]. These stages are shown in Figure. 8 as follows:



**Figure. 8: Knowledge Engineering [41]**

Parts of the literature review performed and subsequent approach applied in this thesis is based on the knowledge engineering stage of Figure. 8. The knowledge is semantically modeled using an ontology. As with any other system, validating a KBS is important to evaluate the final solution, but this task is difficult to perform because of the dynamic nature of the system at hand. Different knowledge representations, constantly growing KB, the sheer number of rules in the KB, and the costs involved in evaluation procedures are regarded as some of the main difficulties in validation of KBS [40].

### 2.3.2 Semantics

Semantics is concerned with *meaning* within the field of logic and linguistics. In this context, logical semantics are related to sense and implications while lexical semantics concern the meaning of the words and their relationships [42]. Linguists try to study semantics to find the general rules and the relationships between the words. For this to be successful, the data (e.g., words) need to be structured in an organized way following linguistic rules, known as syntax. In the domain of information systems, semantics are very useful when large amount of data are available. Structuring these data would allow a reasoning mechanism (e.g., an inference engine) to understand the underlying data and infer and add new knowledge to them.

### 2.3.3 Ontology

The term “Ontology” originates from the field of philosophy, in the early 17<sup>th</sup> century [43], and deals with presence of entities, their similarity and differences, relationships, and categorizing them in a hierarchy, amongst other goals. Perhaps one of the most used descriptions of ontology was provided by Thomas Robert Gruber who defined an ontology as “*an explicit specification of conceptualization*” [44]. By formally defining the objects and associations in the domain of interest, using a representational terminology,

a KB can represent this knowledge. knowledge engineers may create ontologies collaboratively, thus handling the relationships between different ontologies becomes one of the main objectives, especially in a context of modern networked systems. One type of such collaborations could be the dependency of ontologies on one another based on their underlying concepts and relationships [45].

Ontologies are beneficial for reuse of existing knowledge (structuring and organizing domain information), communication within computational systems, humans, and between them, and for computational inferences (evaluating algorithms, inputs/outputs of systems, representing and manipulating plans) [46].

### **2.3.4 Semantic Web**

The SW is defined as a web of data, in general any data obtained such as numbers, properties, dates, and etc. These data represent resources in SW. Following the concept of linked data, by assigning URI's to each resource and using open standards such as RDF, one can organize these data. In the field of SW, Web Ontology Language (OWL) can be used to create vocabularies (or ontologies), SPARQL is used as standard query language, Semantic Web Rule Language (SWRL)[47], is used as the language for adding semantic rules to SW, and reasoning over this data is done by rule based inference engines [48]. The following sections, explain some of these concepts and technologies in greater detail.

### **2.3.5 Web Ontology Language**

The OWL language was created by the W3C and is utilized for creating vocabularies for representing information about entities and their relationships, also referred to as ontologies. In the context of SW, ontologies play a vital role in process automation and OWL as an ontology language had to be positioned into the concept of SW alongside established languages such as Extensible Markup Language (XML) and Resource Description Framework (RDF) [49]. OWL is built on top of the capabilities of RDF and RDF schema (RDF-S), both part of the SW stack of technologies. While RDF [50] allows simple fact stating or flexible representation of information, RDF-S [51], enables class and property structuring, and adding further characteristics to the description of resources. OWL extends these capabilities for example by allowing logical groupings of classes (such as union and intersection) and giving the ability to add more features to properties (such as declaring them symmetric or transitive), amongst other features.

As previously discussed, one of the key advantages of ontologies is that they enable reusing of existing information and OWL has the built in capability to import external ontologies (using owl:imports), allowing the knowledge engineers to merge concepts and relationships between the importing and the imported ontologies [45].

One of the aspects to consider while working with OWL is the fact that it only supports monotonic inferences. This means that the status of the facts and statements defined in OWL will not change if the KB is modified, i.e., new knowledge is added to the KB [52]. For instance, if it is asserted that an individual  $Y$  is an instance of  $B$ , adding more information to the KB will not make this assertion false.

### 2.3.6 Reasoning

Reasoning is one of the fundamentals in the field of KR&R, allowing the inference of implicit knowledge from the explicit knowledge already asserted in the structured knowledge of a domain. This is done by formal manipulation of the symbols demonstrating assumed propositions of the knowledge to create representation of new ones. Such symbols should be particular enough for manipulation (to be taken apart, copied, stringed together) for creating new propositions [4]. For instance, the sentences “Tom loves Jane” and “Jane is coming to the gathering” can be manipulated and result in the sentence “Someone tom loves is coming to the gathering”. This form of reasoning is called “logical inference”.

Knowledge representation formalisms [4], such as ontologies, provide the vocabulary for structuring the knowledge. The underlying semantics of such formalisms have an impact on the reasoning mechanism, hence defining a richer logical relationship amongst the entities of a model, provides a more solid basis for reasoning. This is furthermore increased by addition of use case specific rules, that expand the KB with particular relationships of concepts in any domain. This in turn leads to the increased reasoning capabilities over a KB.

### 2.3.7 Semantic Web Rule Language

The SWRL rule language, increases the expressivity of OWL by adding horn-like rules to OWL axioms. The rules are expressed from the OWL concepts such as classes, properties, individuals, and etc. [53][54]. A reasoner can use the SWRL rules inferring more implicit knowledge from the asserted knowledge. In a simpler view, these rules can be considered as common IF-THEN rules, consisting of an antecedent and a consequent, stating that if and when the antecedent part of a rule is true, then the consequent part should also be true. The antecedent and consequent are composed of conjunctions of OWL predicates and their respective atoms. For instance, the subsequent rule states that if a person has a mother, and the mother has a brother, then the person has an uncle.

$$Person(?x) \wedge hasMother(?x, ?y) \wedge hasBrother(?y, ?t) \rightarrow hasUncle(?x, ?t)$$

The individual discussed above as  $x$  is of type *Person* and has an object property *hasMother*, while the rule itself shows two more object properties. Considering that these concepts have already been defined in an ontology, the addition of such rules al-

allows the reasoning mechanism to infer additional case specific knowledge that cannot be otherwise inferred based on the asserted knowledge. Additionally, SWRL has custom built-ins that greatly increase the expressivity of the rules themselves. These built-ins are categorized as Comparisons, Math, Boolean value, Strings, Date, Time and Duration, URI's, and Lists [47]. The following rule illustrates a SWRL rule extended with one of the comparison built-ins in order to show that any person, who has an age greater than 18, should also be identified as an adult.

$$Person(?x) \wedge hasAge(?x, ?y) \wedge swrlb:greaterThan(?y, 18) \rightarrow Adult(?x)$$

The SWRL built-ins are identified using the *swrlb* namespace. These built-ins can also be used to bind values to arguments. For instance, in *swrlb:add(?x, 4, 6)*, considering that the variable *x* is not bound, after successful execution of this rule, the value 10 will be assigned to variable *x*. It should be noted that like OWL, SWRL supports only monotonic inferences. For example, the next rule takes the antecedent of the previous rule and states that if a person's age is greater than 18, they can participate in a driving test.

$$Person(?x) \wedge hasAge(?x, ?y) \wedge swrlb:greaterThan(?y, 18) \rightarrow canParticipate(?x, true)$$

The successful execution of this rule will add the *canParticipate* property with status true to the individual. However, if the individual already held the same property with a false status, the individual will have two values for the same property after the rules execution.

SWRL rules allow knowledge engineers to address different knowledge management aspects. For example, syntactic mapping of elements in the process of transformation between distinctive knowledge models could result in semantic loss, which can be enhanced with the use of rules [55]. It is possible to add constraints to the structural knowledge of an application, further enhancing the characteristics of properties [56]. Modelling the problems and solutions of a particular domain, in form of antecedents and consequents, respectively, allows inferring additional application specific knowledge as previously discussed [57].

### 3. METHODOLOGY

The state of the art review of concepts and technologies available for the proposed solution were described in the previous chapter. Based on that, this chapter aims to demonstrate the approach used for this thesis work, alongside the methods and tools selected.

#### 3.1 Manufacturing model

The proposed manufacturing systems model, which will be explained in more detail in the following sections, has been modelled using the ISA-95 standard. Because of the sheer number of elements and concepts available in the standard that can be used for manufacturing systems, the functional hierarchy model has been used to help categorizing the functional elements of the planned model. This categorization leads to the identification and ultimately creation of three different ontologies, defined as sub-ontologies here in after. Each of the sub-ontologies defines a distinct underlying functional group of the proposed model. As previously explained, importing external ontologies, allows the merging of concepts and relationships. This process involves utilizing the sub-ontologies as imports in an empty ontology and saving the new ontology as the final envisioned manufacturing systems ontology, known as the Enterprise Control Ontology (ECO), in this thesis work.

The de-modularization of the underlying functional groups as sub-ontologies should allow knowledge engineers to create domain or application specific ontologies, with distinct entities and relationships. This becomes particularly useful when the different sub-ontologies are not modelled by the same engineers and hence facilitate the possibility to use any third-party external model for an implementation as long as it has the same purposes [58]. However, it should be taken into consideration that ontologies that employ imports have certain characteristics and utilizing them demands more attention to particular aspects from the user. One of such aspects will be the existence of several different namespaces that should be considered when querying the model.

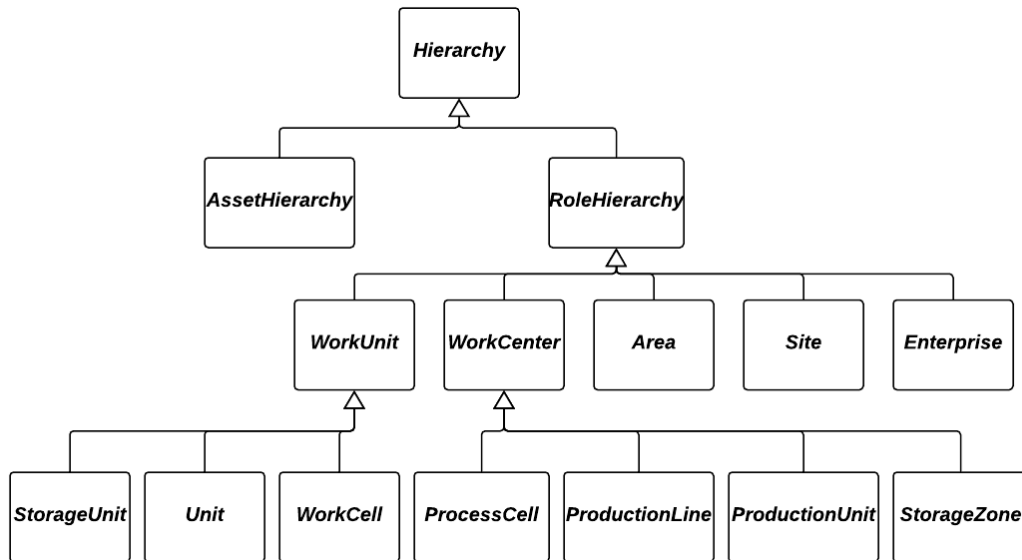
##### 3.1.1 Hierarchy sub-ontology

The hierarchy sub-ontology is designed to illustrate the assets of the enterprise involved in manufacturing based on the Role-based Hierarchy model and the Physical Asset Equipment model of part one of the standard [25]. The former identifies the assets based on their functionality, while the latter identifies them based on their physical composition, location, possible relationship with other resources, or financial aspects. This dis-



tion has been used to present every asset as a subclass of *RoleHierarchy* or *AssetHierarchy*, both modelled and written in shorter forms to allow easier readability. Assets can be defined from both aspects as well, as elements from these two models may correspond at certain levels as previously shown in Figure. 5. The standard further defines the areas of responsibility in a role-based model into *Enterprise*, *Site*, *Area*, *WorkCenter*, and *WorkUnit*.

Work centers were described as equipment elements categorized under an area and have particular terminologies for each production type, which is also reflected in the *WorkCenter* subclasses, *ProcessCell* (batch production), *ProductionLine* (Discrete production), *ProductionUnit* (Continuous production), and *StorageZone* (material/equipment storage and movement). Work centers are made up of work units, the lowest level of equipment in role-based hierarchy model. The classification of *WorkUnits* is demonstrated in subclasses as *StorageUnit*, *Unit*, and *WorkCell*. All of the above-mentioned classifications are shown in Figure. 9.



**Figure. 9:** *Hierarchy sub-ontology model*

ISA-95 part two defines application independent attributes and object properties used for the elements of part one, e.g., assets of the enterprise in role-based or physical asset hierarchy models. In the scope of this work, the properties considered for the hierarchy sub-ontology are presented in the next section.

### 3.1.2 Properties of hierarchy sub-ontology

Some of the object and data properties used within the hierarchy sub-ontology are demonstrated in Table. 3, completed with additional information about the function of the mentioned properties and the domain and ranges assigned.

Properties can be thought as the relationship between two entities, a subject and an object. The domain and range features identify the type of the individuals that can be selected as the subjects and objects in that relationship, respectively. For instance, a relationship using the *contains* object property below, could be modelled as *StorageZoneX contains StorageUnitX*. In the mentioned example, *StorageZoneX* and *StorageUnitX* are both of type *RoleHierarchy*. It should be additionally noted that any individuals selected as the subjects and objects of a relationship will be inferred as a type of the domain and range specified for the relationship.

**Table. 3:** *Properties of hierarchy sub-ontology*

No.	Property type	Name of Property	Domain	Range
1	Object Property	contains	RoleHierarchy	RoleHierarchy
2	Data Property	hasDescription	owl:Thing	xsd:string
3	Data Property	hasEquipmentCapabilityType	RoleHierarchy	xsd:string
4	Data Property	hasEquipmentID	WorkUnit	xsd:string
5	Data Property	hasEquipmentLevel	RoleHierarchy	xsd:string

The first property, i.e., *contains*, aims to assign role-based subordinate equipment to their higher-level equipment. For instance, an instance of *Area* can contain any number of *WorkCenter* instances. The *hasDescription* property, describes additional information about the property and can be assigned to any type of individual. The *hasEquipmentCapabilityType* property, defines the capability type of the equipment used, which can be “Available”, “Committed”, “Used”, “Unused”, “Total”, or “Unattainable”. As it can be seen in the table above, the property defined for the identification of an equipment, i.e., *hasEquipmentID*, is following the naming scheme provided in the standard for role-based equipment. The same approach has been performed for the implementation of other elements whenever the standard clearly distinguishes the categories of elements. The *hasEquipmentLevel* property, categorizes the equipment based on their level in role-based hierarchy. For example, an instance titled *Welder*, can be of level *WorkCenter*.

### 3.1.3 Operation Type sub-ontology

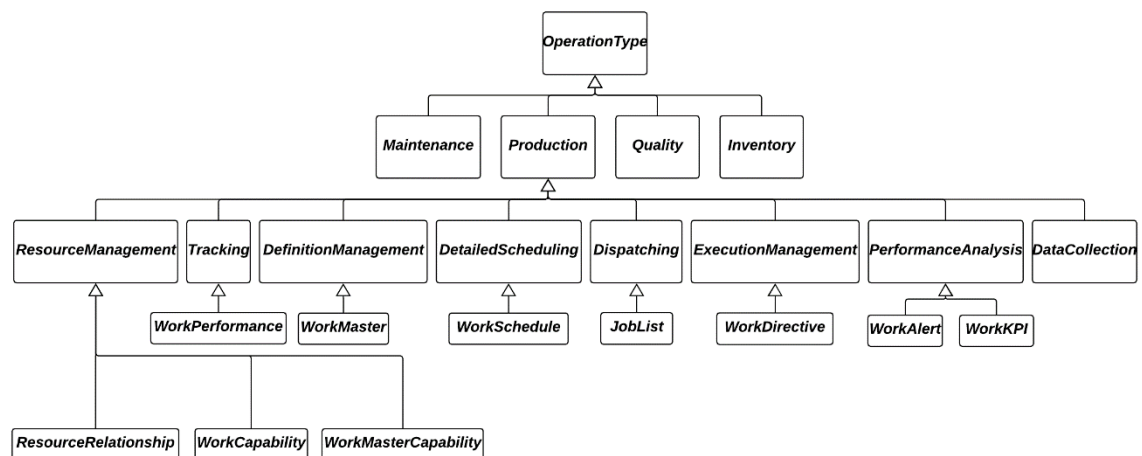
The operation type sub-ontology is based on the MOM level (equivalent to the MES level) and defines a set of activities that coordinate the equipment, material, personnel, and energy, to perform specific operations [27]. These operations are categorized under four different management activities as, *production*, *maintenance*, *quality*, and *invento-*



- Work schedule: Comprised of a set of job orders to be executed in the production and their corresponding sequencing. Work schedules are created by the detailed scheduling activity.
- Job list: Defines the set of job orders to be performed at specific work centers or work units and are created by the dispatching activity.
- Work performance: Can be defined as a set of work responses from the manufacturing activities associated with a work request (set of job orders). Work performances created within the tracking activity.
- Work alert: Can be created by any of the activities of MOM and may be triggered by certain event occurrences.
- Work KPI: Measurable performance indicators having operational value for the enterprise. Work KPIs are created by analysis activities.
- Work capability: Certain grouping of resources with specific capabilities required for performing stages of a work definition. Work capabilities are created by the resource management activity.
- Work master capability: The ability of resources to execute tasks and their capacities. Work master capabilities are created by the resource management activity.
- Resource relationship network: Can be defined as expressions of relationship between resources and are formed by assignments in resource and definition management.

In the resource sub-ontology, work models are created as *WorkMaster*, *WorkDirective*, *WorkSchedule*, *JobList*, *WorkPerformance*, *WorkAlert*, *WorkKPI*, *WorkCapability*, *WorkMasterCapability*, and *ResourceRelationship*, each as subclasses under their corresponding category.

Figure. 11 shows the complete operation type model.



**Figure. 11:** Operation Type sub-ontology model

### 3.1.4 Properties of operation type sub-ontology

The attributes and object properties for operation type ontology are selected from the 4<sup>th</sup> part of the standard that details the object models and attributes of MOM [28]. In the scope of this work, some of the attributes selected are shown in Table. 4.

*Table. 4: Properties of operation type sub-ontology*

No.	Property type	Name of Property	Domain	Range
1	Object Property	CorrespondsToWorkSchedule	WorkPerformance	WorkSchedule
2	Object Property	referencesWorkMaster	JobOrder	WorkMaster
3	Data Property	hasJobListID	JobList	xsd:string
4	Data Property	hasPriority	JobOrder	xsd:string
5	Data Property	hasPublishedDate	WorkSchedule	xsd:string
6	Data Property	hasWorkMasterCapacityType	WorkMaster	xsd:string
7	Data Property	hasWorkPerformanceID	WorkPerformance	xsd:string
8	Data Property	hasWorkScheduleID	WorkSchedule	xsd:string
9	Data Property	hasWorkType	OperationType	xsd:string

The *CorrespondsToWorkSchedule* property, identifies the associated work schedule for a work performance. The *referencesWorkMaster* property, identifies the associated work master for a job order. The *hasJobListID* property, defines a unique identification of the job list. The *hasPriority* property, identifies the priority of job orders that could be for instance “Highest”, “3”, “A”, or “Medium”, all depending on the definition of priority type. The *hasPublishedDate* property, identifies the date and time in which a work schedule was generated. The *hasWorkMasterCapacityType*, identifies the capacity for work masters, which could be “Available”, “Committed”, or “Unattainable”. The *hasWorkPerformanceID* and *hasWorkScheduleID* properties, describe unique identifications for work performance and work schedules, respectively. And finally, the *hasWorkType* property, identifies the category of work, which could be “Production”, “Maintenance”, “Inventory”, or “Quality”.

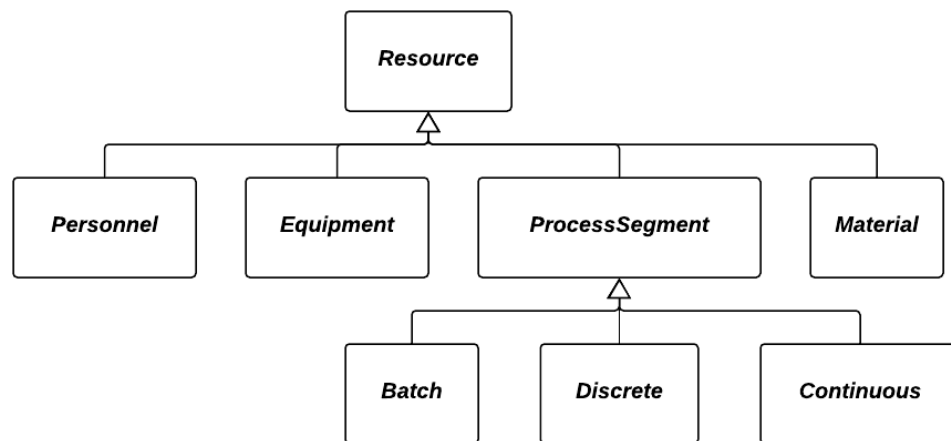
### 3.1.5 Resource sub-ontology

The resource sub-ontology is the third and final sub-ontology in this work. ISA-95 describes resources as entities providing the capabilities needed for performing the enterprise and/or business activities and processes. The resources involved in manufacturing are described as follows:

- **Personnel:** The personnel involved in MOM, defined in the *Personnel* class.
- **Material:** The materials used in MOM including raw, intermediate, and finished materials, and consumables, defined in the *Material* class.
- **Equipment:** The main equipment involved in manufacturing, defined in the *Equipment* class. In the context of this work, the *Equipment* class is representing the assets based on the physical asset hierarchy model. The role-based equipment is defined as in the *RoleHierarchy* class.
- **Process segment:** Resources with detailed functionalities required for a segment of production. The segments of production discussed for process segments could be of production, maintenance, inventory, or quality type. Process segments in the ontology are defined as *ProcessSegment*.

Work masters (as discussed in the previous section) and process segments both define a view of the manufacturing processes, but seen from different levels namely, level three and four, respectively. This is another example of how the standard uses the functional hierarchy model in distinguishing the enterprise and control/manufacturing processes. In this sense, process segments are more business oriented.

The categorization of resources is shown in Figure. 12. As it can be seen the process segment has been further extended with *Batch*, *Discrete*, and *Continuous* production types.



**Figure. 12:** Resource sub-ontology model

### 3.1.6 Properties of resource sub-ontology

The attributes and object properties for resource ontology are selected from the second part of the standard [26] that details the object models and attributes for the elements of the enterprise-control integration. In the scope of this work, some of the selected attributes are shown in Table. 5.

*Table. 5: Properties of resource sub-ontology*

No.	Property type	Name of Property	Domain	Range
1	Object Property	RequiresMaterialDefinition	ProcessSegment	Material
2	Object Property	RequiresPersonnel	ProcessSegment	Personnel
3	Object Property	RequiresPhysicalAsset	ProcessSegment	Equipment
4	Data Property	hasAssemblyRelationship	Material	xsd:string
5	Data Property	hasAssemblyType	Material	xsd:string
6	Data Property	hasPhysicalAssetCapabilityType	Equipment	xsd:string
7	Data Property	hasOperationType	ProcessSegment	xsd:string
8	Data Property	hasPhysicalAssetID	Equipment	xsd:string
9	Data Property	hasPhysicalLocation	Equipment	xsd:string
10	Data Property	hasProcessSegmentID	ProcessSegment	xsd:string
11	Data Property	hasVendorID	Equipment	xsd:string

The *requiresMaterialDefinition* property, identifies the material needed for a process segment. Instances of material could be “city water” or “grade B aluminum”. The *RequiresPersonnel* property, identifies the personnel needed for a process segment. The *RequiresPhysicalAsset* property, identifies the physical asset needed for a process segment. The *hasAssemblyRelationship* property, defines the type of relationship as “Permanent” when an assembly is not planned to be split during production process, or as “Transient” representing the use of the material as a temporary assembly, such as a pallet. The *hasAssemblyType* property, identifies the type of an assembly, as “Physical” when the modules of the assembly are physically connected or exist in the same area, or as “Logical”, if the modules are not connected or in the same area. The *hasPhysicalAssetCapabilityType* property, defines the capability type of a physical asset used as “Available”, “Committed”, “Used”, “Unused”, “Total”, or “Unattainable”. The *hasOperationType* property, defines the category of operation for a process segment as “Production”, “Maintenance”, “Quality”, “Inventory”, or as “Mixed” when the activity corresponds to several categories. The *hasPhysicalAssetID* and *hasVendorID* properties, both define identifications for a physical asset. The former assigns a unique enterprise wide identification to the asset while the latter can be used to check vendor information if required. The *hasPhysicalLocation* property, defines the actual physical location of the equipment. And finally, the *hasProcessSegmentID* property, defines a unique identification for process segments.

In the final envisioned ECO model, the resource ontology is seen as an intermediary level, between the hierarchy and operation type sub-ontologies explained. Therefore, it

is necessary to consider the required intra-level mappings to facilitate proper information exchange. This can be achieved by adding the linking attributes and object properties defined in parts two and four of the standard, to the ECO model.

### 3.2 Ontology editor

The reuse of knowledge is one of the main benefits of using ontologies. This becomes even more apparent in the context of a more networked and dynamically changing world, where many applications may share the same ontology. But the number of suitable and available ontologies is not similar in different fields. Hence, ontology editors enable users to create their own ontologies by defining the entities, and relationships corresponding to their use case and domain of interest.

The Protégé<sup>3</sup> ontology editor is one of the well-established applications of such that was developed by the Stanford Center for Biomedical Informatics Research at the Stanford University School of Medicine. Protégé provides the users with a simple application interface, that can be customized based on user's preferences, supporting the creation of ontology axioms, e.g., classes, object properties, etc. Additionally, it provides visualization tools for navigating and analyzing the relationships. support aids for explaining inconsistencies, and an interface to connected reasoners, amongst other features [59][60].

### 3.3 Remarks of approach

The terminologies and taxonomies available in ISA-95 provides engineers with a uniform naming scheme to model and implement their desired elements in an industrial scenario. It was one of the main goals of this chapter to make the standard provided guidelines as clear as possible in the modelling of every sub-ontology. Additionally, the properties selected and demonstrated aim to present examples for any user who intends to use the ECO model as a basis for their implementation and extending them. Another goal of the approach is to provide a general view of possible properties that correspond to different elements in the domain of manufacturing, such as equipment, material, personnel, and etc., even though all may not be used in specific parts of the implementation.

---

<sup>3</sup> <https://protege.stanford.edu/>



## 4. IMPLEMENTATION

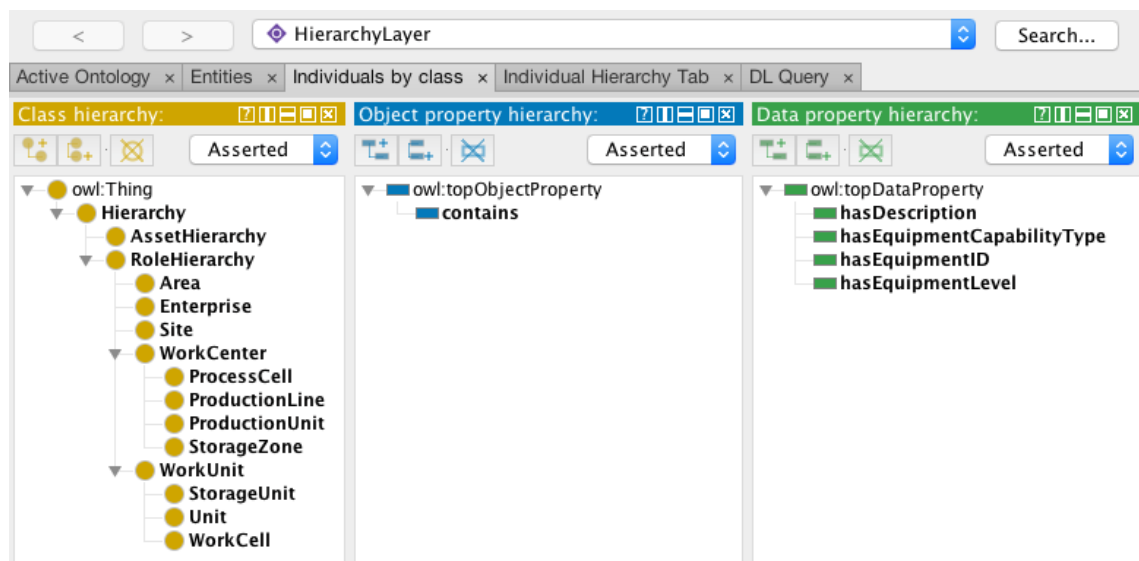
The following chapter defines the implementation for realizing the final envisioned manufacturing model, i.e., ECO, its instantiation within the use case presented, and demonstrating the supporting semantic rules in two subsections. In the following chapter, ECO has been developed using the Protégé ontology editor.

### 4.1 Implementation of manufacturing systems model

The Enterprise Control Ontology (ECO) is created by importing three sub-ontologies in an empty ontology and merging their class definitions, object and data properties. This approach offers a certain level of abstraction in the design of the sub-ontologies that becomes particularly useful when there are incorrect or incoherent (in the context of the standard used) concepts that need addressing at the lower sub-ontology levels. Changes made and saved in the underlying sub-ontologies will automatically transfer to the ECO model, as intended without any discrepancies.

#### 4.1.1 The Hierarchy sub-ontology implementation

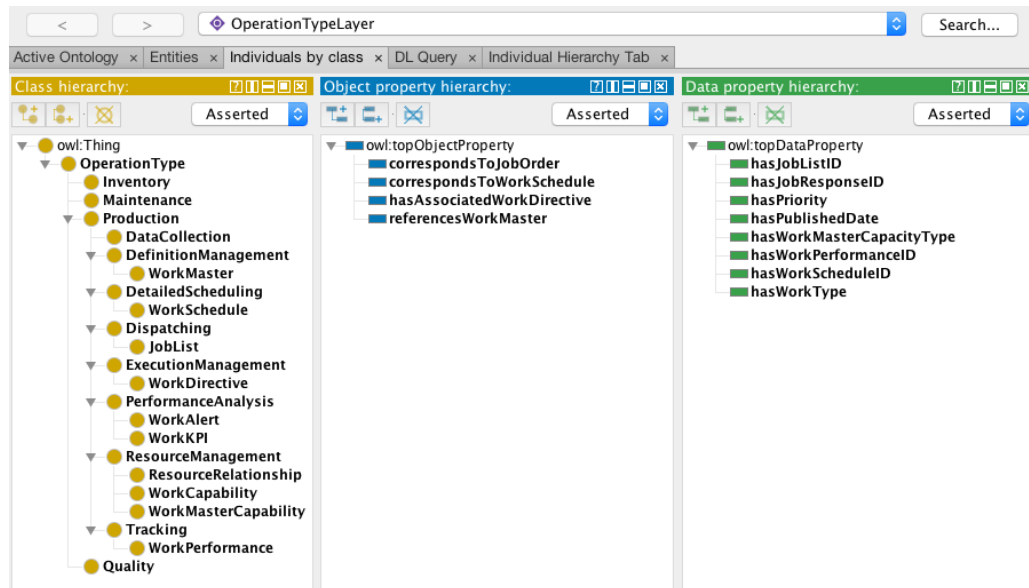
The hierarchy sub-ontology defines the main assets of the enterprise based on the role-based and physical asset hierarchy models. Figure. 13, shows the class, object, and data property hierarchies of the sub-ontology.



*Figure. 13: Hierarchy sub-ontology implementation*

### 4.1.2 The Operation Type sub-ontology implementation

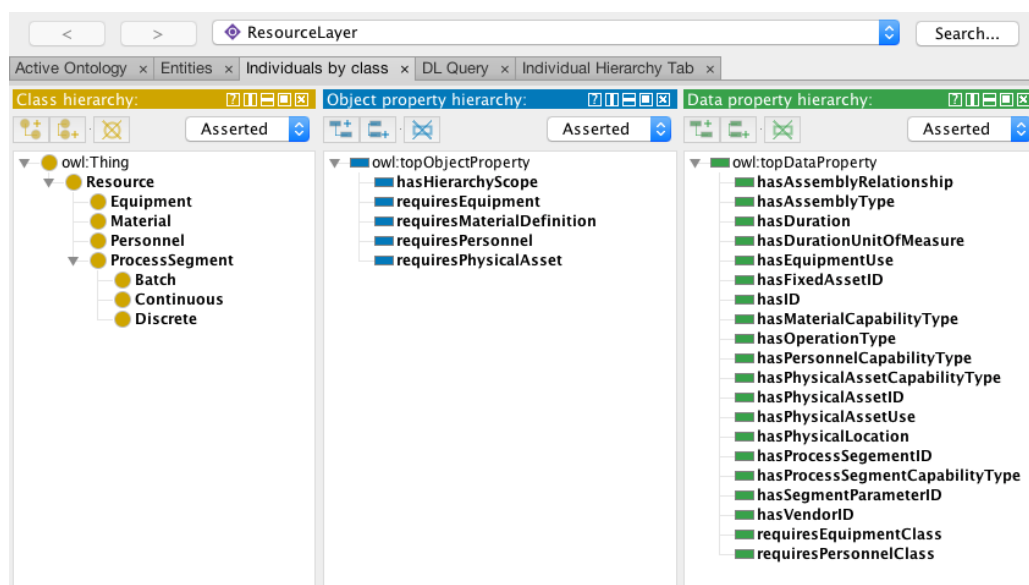
The Operation Type sub-ontology defines manufacturing operation activities within the MOM level. Figure. 14, shows the class, object, and data property hierarchies of the sub-ontology.



*Figure. 14: Operation Type sub-ontology implementation*

### 4.1.3 The Resource sub-ontology implementation

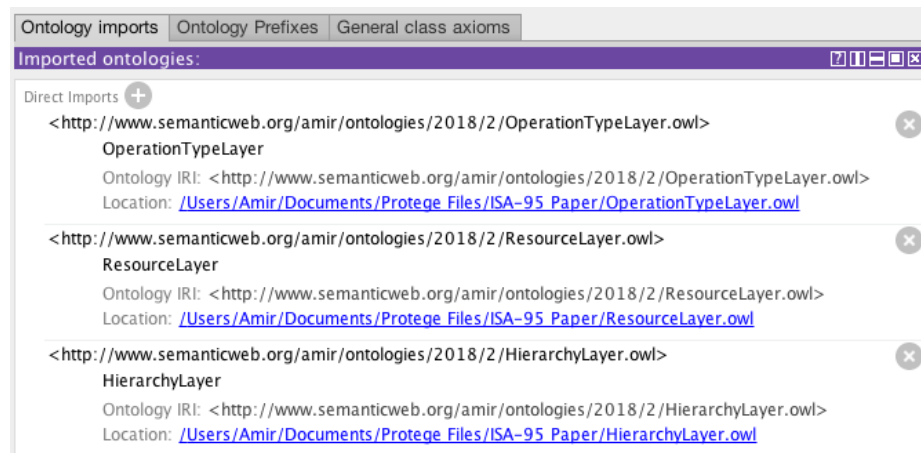
The resource sub-ontology defines resources involved in manufacturing operation activities within the enterprise. Figure. 15, shows the class, object, and data property hierarchies of the sub-ontology.



*Figure. 15: Resource sub-ontology implementation*

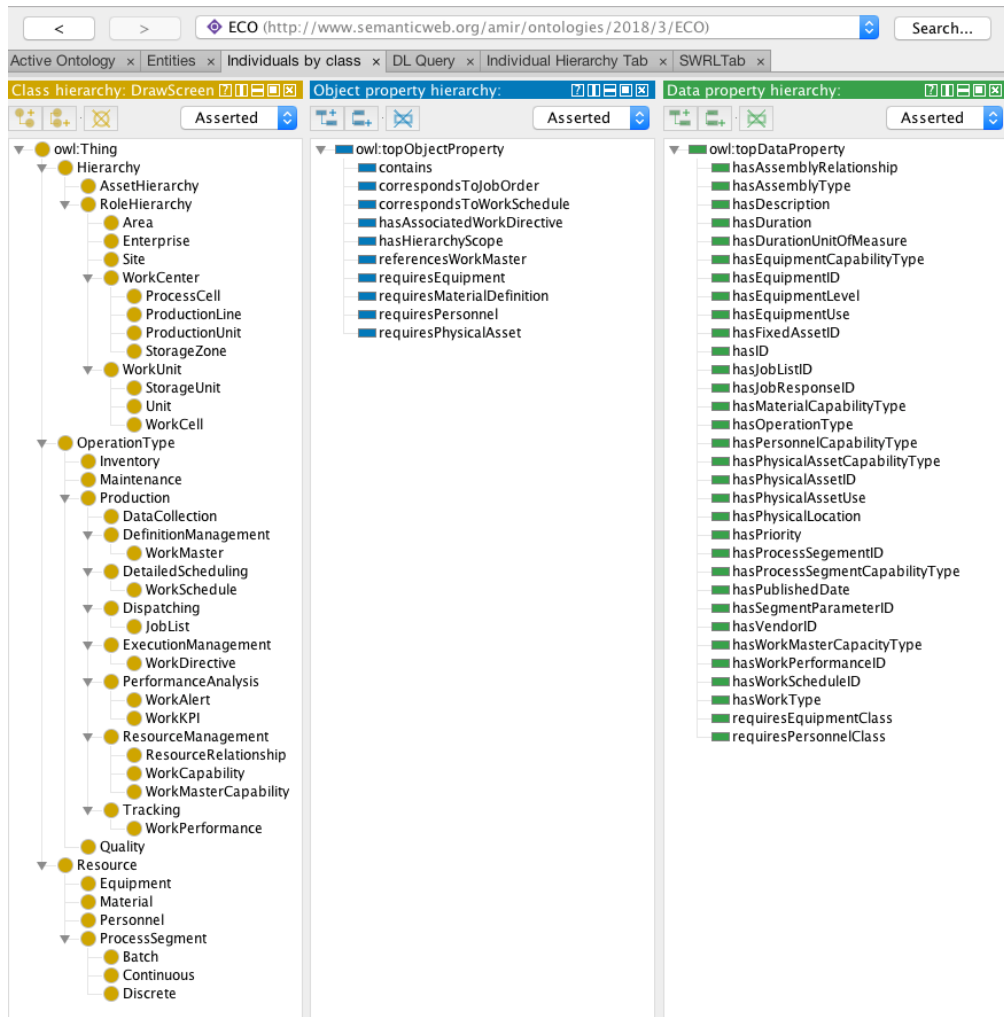
#### 4.1.4 Enterprise Control Ontology (ECO)

The importing of the sub-ontologies is performed through the “Ontology imports” tab of Protégé. The process allows users to import an ontology contained in a specific file, a document located on the web (by providing the ontologies URL), loading it from the workspace, or importing from any of the ontology libraries available. In the case of this implementation, the sub-ontologies were imported from specific files on the designated directory. It should be noted that the sub-ontologies have to be present in the same directory, as shown in Figure. 16, and additionally, the new ontology should also be saved in the same directory. This will ensure proper transfer of changes in the underlying sub-ontologies to the ECO model.



*Figure. 16: Sub-ontology imports*

The successful import process results in the aggregation of the class, object, and data properties of the sub-ontologies as shown in Figure. 17. In this stage, case specific entities and relationships (called collectively as axioms) can be added to the ECO model. These new additions will be highlighted in bold in any chosen hierarchy, making it easier for the user to spot case specific concepts that are added as extensions. Additionally, one can simply check the namespace of every axiom to identify the corresponding ontology.



**Figure. 17:** Enterprise Control Ontology (ECO) implementation

## 4.2 Use case implementation

This section details an extension to the implemented ECO model explained in the previous section, by instantiating the model as part of an industrial use case, for providing a proof of the concept. The description of the use case paves the way for a better understanding the instantiating process. In the end, the supporting semantic rules employed for linking the concepts and achieving the targets set for the use case will be discussed. It should be taken into consideration that it is not the intention of utilizing the use case for demonstrating a fully-fledged manufacturing system, but to provide insights into the potentials of the approach presented.

### 4.2.1 FASTory line

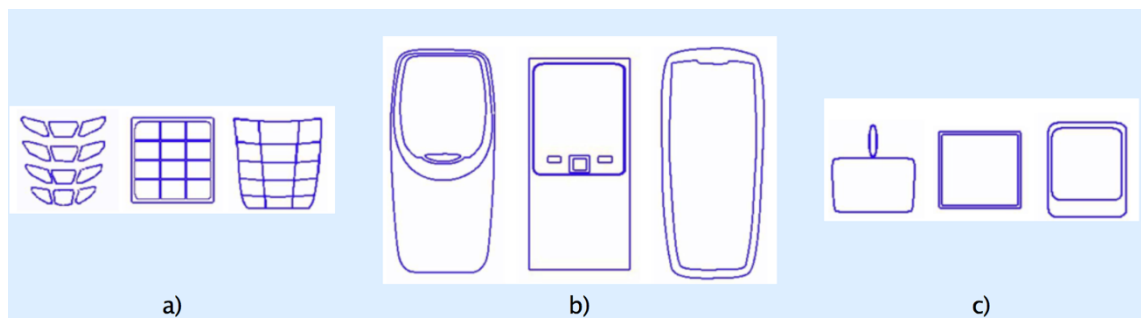
The FASTory line, as shown in Figure. 18, is a discrete assembly line located in the Factory Automation Systems and Technologies Laboratory (FAST-Lab), part of the

Tampere University of Technology (TUT). The available equipment in the assembly line provide a good opportunity for user's interaction with real industrial equipment.



**Figure. 18:** *FASTory assembly line*

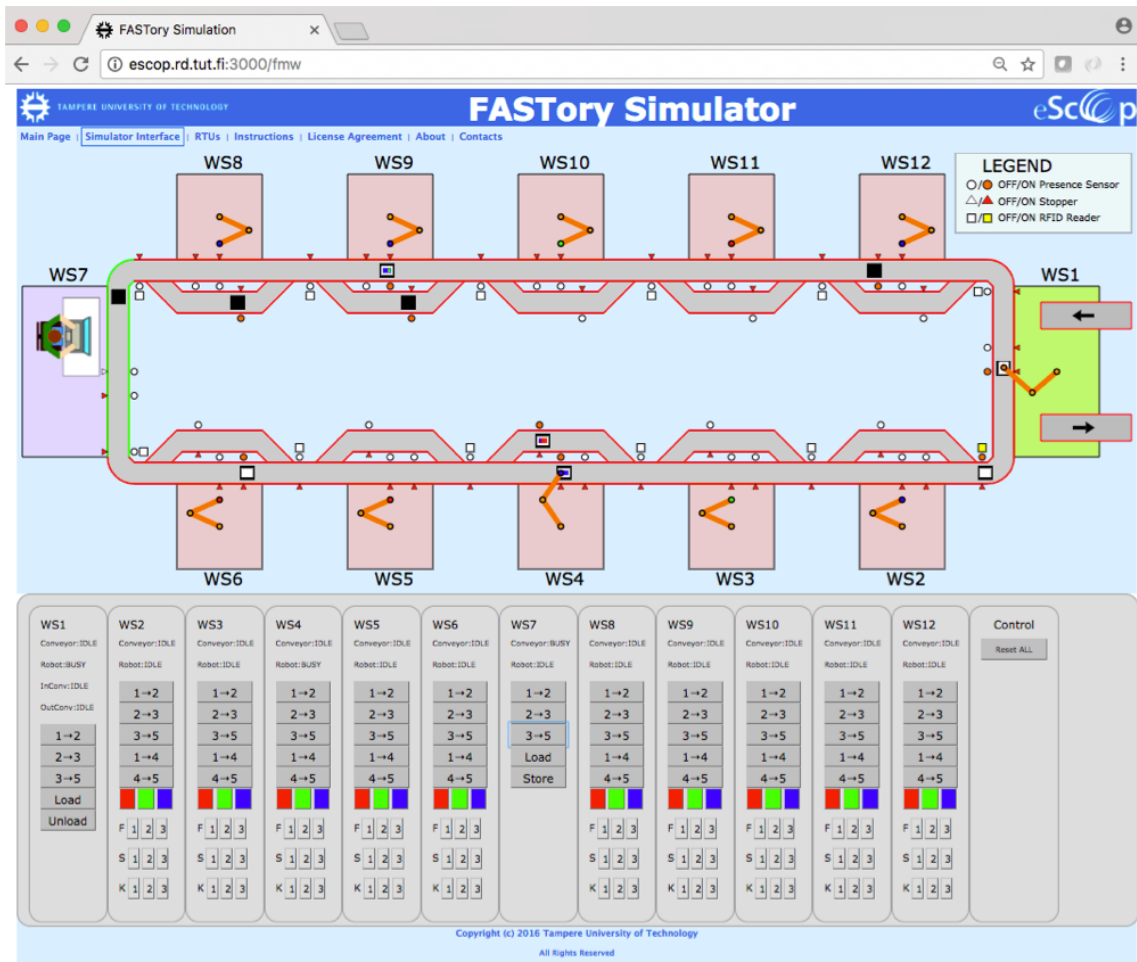
The FASTory line displays assembly of mobile phones by drawing the screen, keyboard, and frame of mobiles on pieces of paper that are loaded on pallets [60]. In the context of this implementation, the word “assembly” refers to specific “drawings” of mobile components. Each assembly (frame, screen, keyboard) has three distinct patterns, which are shown in Figure. 19.



**Figure. 19:** *a) Keyboard patterns b) Frame patterns c) Screen patterns [61]*

The configuration of the line is as follows. Workstation 1, is equipped with a SCARA robot and is in charge of loading and unloading papers onto pallets. Workstation 2-6 and 8-12 are all also equipped with SCARA robots. The main phone assembly operations are performed in these stations. In practice, each station is capable of drawing all possible assembly patterns. There is also a pen feeder that allows the robot to pick the available pen colors such as blue, red, and green, for instance. And finally, the task of workstation 7 is the loading and unloading of pallets into the FASTory line. Based on the configuration, the operation of workstation 7 can be manual or automatic.

Each assembly workstation employs a double path conveyor. The main path, takes the pallets to the assembly zone while the secondary path is meant as a bypass line. This feature is necessary to handle the pallets traffic loads and avoid blockage, particularly useful when the station is not active for operation or it is indeed performing assembly operations. The arrangement of work stations and conveyors is shown in more detail in Figure. 20.



*Figure. 20: FASTory simulator [60]*

#### 4.2.2 General FASTory equipment instantiation

The FASTory line is designed and operated as a fully automatic assembly line. Hence in such an assembly line, the role of human/user activity is minimum to none. Some of the limited interaction of personnel with the line might include loading and unloading pallets, in case workstation 7 is configured as manual, and performing quality and maintenance tasks. For the purpose of this instantiation, the utilization of personnel has not been included. The main equipment, resources, and operations used in the line are explained as follows:

- Workstations 1-12

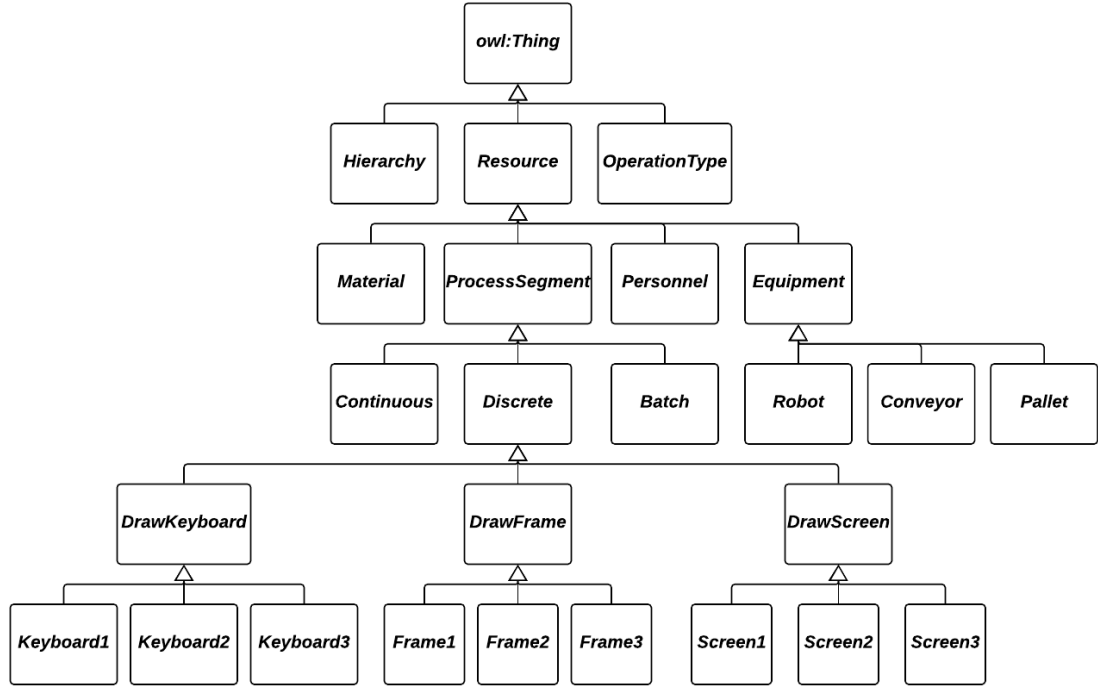
- SCARA robots 1- 6 and 8-12 (workstation 7 is considered as manual)
- Assembly operations: Detailing every pattern of frame, screen and keyboard assembly)
- Zone transfer: The zone transfers present in every workstation
- Pallets

Instances of classes, known as individuals from now on, are added in Protégé within the individuals tab, which allows declaring the type of the individual and its data and object properties, amongst other features. The adoption to the standards uniform naming schemes and taxonomies is considered throughout this implementation and will be explained where ever necessary. Workstations are one of such examples. The standard defines the term “work unit” for the lowest levels of equipment in the role-based hierarchy. This is additionally further categorized as “work cell” in discrete manufacturing operations. Hence, the naming used for the instantiation of workstations is changed into “work cell” and will be used from here on. The work cells are abbreviated as *WC*, as a type of *WorkUnit*, and subclass of *RoleHierarchy*. Twelve work cell individuals are instantiated with their corresponding abbreviations as *WC1*, *WC2*, and etc.

The *equipment* class of ECO (from the resource sub-ontology) is further extended with three subclasses as *Robot*, *Conveyor*, and *Pallet*, resulting in a more detailed classification of any equipment. On this account, equipment is referring to the physical assets involved in manufacturing (described in section 2.2.5). Any type of equipment should also be of type *AssetHierarchy*. The method used for defining this type of assignment is later explained as part of a semantic rule. The SCARA robots are defined as *Robot* with their corresponding designations, hence twelve robot individuals are instantiated as *Robot1*, *Robot2*, and etc. Each robot individual is extended with the supplementary data property *hasDesignation* that has the same designation as the name of the robot. Additionally, the status of each robot is shown using the *hasPhysicalAssetCapabilityType* data property. In the case of FASTory implementation the existing statuses are considered as “Available” or “Not-Available”.

The assembly operations of the robots are modelled in *ProcessSegment* class of resources. Hence the terms “process segment” and “assembly operations” may have been used interchangeably here in after. The available patterns for every assembly are instantiated with the abbreviation of the type of operation and their corresponding designations. For doing so, the *Discrete* class, subclass of *ProcessSegment* was further extended with three subclasses named *DrawFrame*, *DrawScreen*, and *DrawKeyboard*. These represent the main categories of assembly operations. Each category, is then extended with another three subclasses, to present specific assembly patters. For instance, the class *Frame1*, subclass of *DrawFrame*, identifies the frame assembly operation with pattern one. The extensions to the *Equipment* and *ProcessSegment* classes are shown in Figure. 21.





**Figure. 21:** Extensions of ECO model

The above-mentioned steps are then followed by the addition of nine individuals, each types of the lowest level of process segments in the class hierarchy. For example, the individuals *K3*, *F2*, and *S1*, are of types *Keyboard3*, *Frame2*, and *Screen1*, respectively, each demonstrating a single assembly operation. These lower level process segments also represent the assembly operations that each robot is capable to perform. Hence, the same individuals are also used for modeling each robot's functionality, which will be explained in more details in the following subsections.

The next addition to the ECO model is the class *JobOrder*, which is created as a subclass of *Dispatching*, within the operation type class hierarchy. Job orders represent the smallest unit of work, defined by work masters, for execution. And finally, as FASTory is a discrete assembly line, the main class *OrderManagement* has been added to the model, and further extended with the subclass *OrderRequests*. The instances of *OrderRequests* demonstrate orders made up of three distinct assembly requests (Frame, screen, and keyboard). Three new supplementary object properties have been added to ECO to model these assembly requests namely, *hasFrameRequest*, *hasScreenRequest*, and *hasKeyboardRequest*. For instance, Figure. 22 shows *Order1*, with *F1*, *S3*, and *K1* assembly requests.

The use of the assembly operations, job orders, and assembly requests will be described in more details within the semantic rules added.



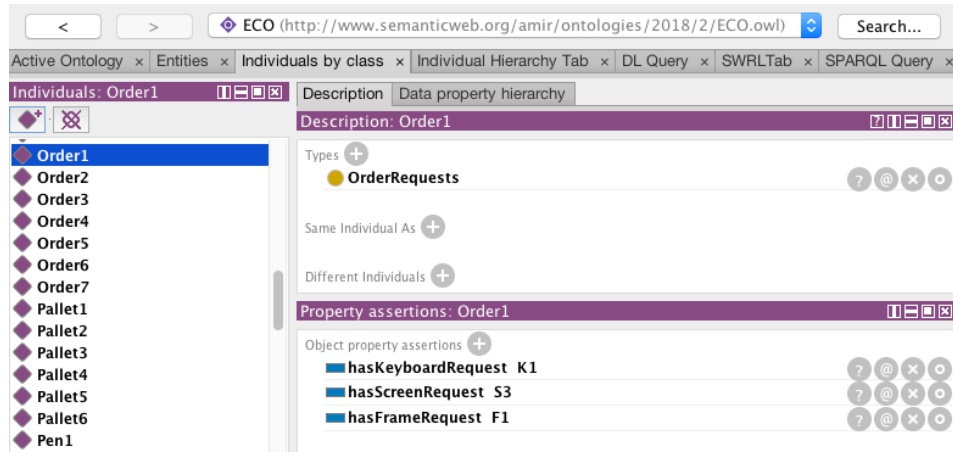


Figure. 22: Order1 assembly requests

### 4.2.3 FASTory zone-transfer instantiation

As previously described, each assembly work cell has a main conveyor path for assembly operations and a bypass path to avoid blockage and handle pallet traffic. Each of the assembly work cells (WC2-6, WC8-12) have five different work zones, while WC1 and WC7, have four and three work zones, respectively. The overall conveyor setup of FASTory is shown in Figure. 23.

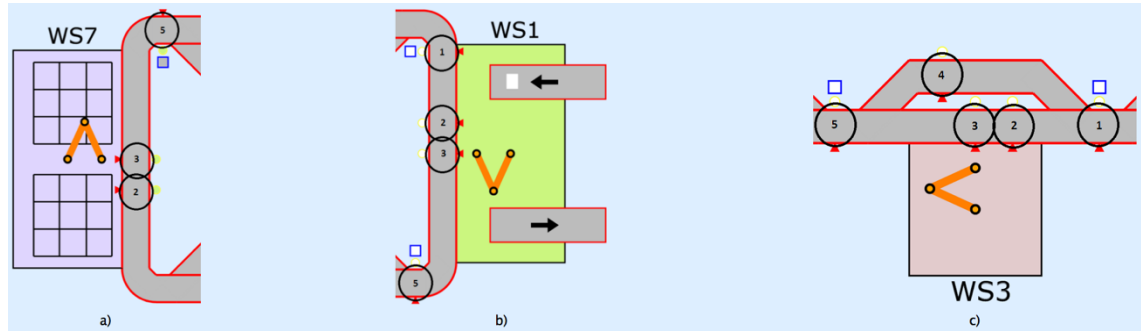


Figure. 23: a) Work zones in WC7 b) Work zones in WC1 c) Work zones in WC2-6 / 8-12 [61]

In order to model the zone-transfer operations, the class *ZoneTransferOP* has been added to the ECO model to allow the creation of operations as instances of the class. It should be noted that new class is positioned with *Hierarchy*, *OperationType*, and *Resource* main classes in the same level of the class hierarchy. Considering a starting (from) and an ending (to) position for each zone transfer within a work cell, there are five possible zone-transfer operations in the assembly work cells, while the WC7 (pallet load/unloading) and WC1 (paper load/unloading), each have two and three zone-transfer operations, respectively. Zone-transfer operations have been modeled as ZT in addition to their corresponding “from-to” zones and work cell designation. Hence fifty-five instances of zone transfer operations have been added to model all possible zone-

transfer operations in FASTory line. For instance, the zone transfers of work cell 5 are shown as follows in Table. 6.

**Table. 6:** *Work cell 5 zone-transfers*

Zone-transfer	Description
ZT12_5	Transfer operation between zone one and two of work cell 5 (Assembly path)
ZT23_5	Transfer operation between zone two and three of work cell 5 (Assembly path)
ZT35_5	Transfer operation between zone three and five of work cell 5 (Assembly path)
ZT14_5	Transfer operation between zone one and four of work cell 5 (Bypass path)
ZT45_5	Transfer operation between zone four and five of work cell 5 (Bypass path)

#### 4.2.4 FASTory work master instantiation

The standard describes work masters as templates with information about the necessary resources and routing for executing a unit of work without referencing an actual job order. In order to define FASTory specific work master types, the *WorkMaster* class of ECO has been extended with the two subclasses *ProductDefintion* and *ZoneTransfer*.

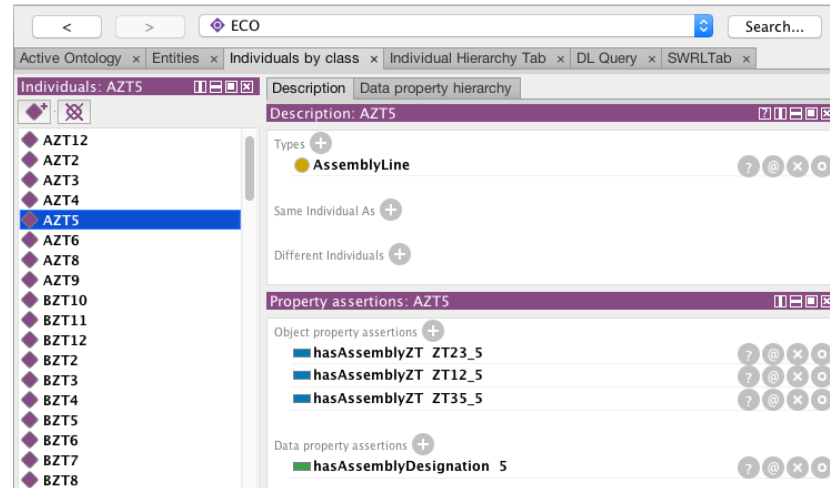
The *ProductDefinition* class defines the type of products the assembly line can produce. With three patterns available for each assembly operation the work masters can represent every product type based on their composition. To do so, *ProductDefinition* is extended with subclasses representing the available types as *Type1*, *Type2*, and etc. The type instances added to the FASTory implementation are named in a manner to clearly represent the product definition such as *F2SIK2*, denoting the frame assembly with pattern two, screen assembly with pattern one, and keyboard assembly with pattern two.

The second subclass of *WorkMaster* is the *ZoneTransfer* class, which is intended to demonstrate work masters for performing a collection of specific zone-transfer operations inside work cells. For instance, the zone-transfer operations ZT12, ZT23, and ZT35 are herein after collectively called “Assembly zone-transfers”. The three main categories of such zone-transfer collections are shown as follows:

- Assembly zone-transfer (AZT): Collection (trio) of ZT12, ZT23, and ZT35 zone-transfers that enable a pallet to transfer within the main conveyor path of WC2-6 and WC8-12
- Bypass zone-transfer (BZT): Collection (pair) of ZT14 and ZT45 zone-transfers that enable a pallet to transfer within the bypass conveyor path of WC2-6 and WC8-12
- Loading zone-transfer (LZT): Collection (trio) of ZT12, ZT23, and ZT35 zone-transfers that enable a pallet to transfer within the loading path of WC1

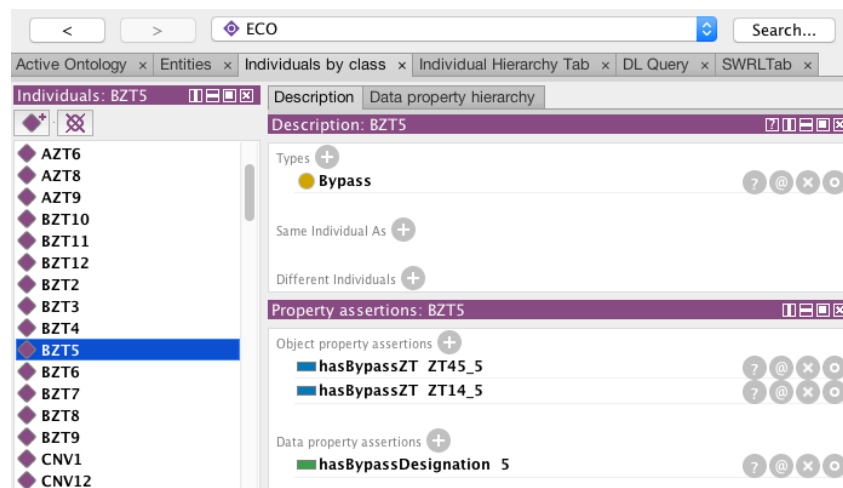
The above-mentioned classification results in the addition of *AssemblyLine*, *Bypass*, and *LoadingLine* classes (subclasses of *ZoneTransfer*) to the model. In the case of AZT, the

*hasAssemblyZT* and *hasAssemblyDesignation* object and data properties are added to ECO as supplementary properties to demonstrate association of zone-transfers. The *hasAssemblyDesignation* holds the same designation as the instance itself. For example, the AZT of work cell five is instantiated as *AZT5* as shown in Figure. 24. This individual is extended with three *hasAssemblyZT* object property assertions to individuals *ZT12\_5*, *ZT23\_5*, and *ZT35\_5* and assembly designation “5”.



**Figure. 24:** Assembly zone transfer 5

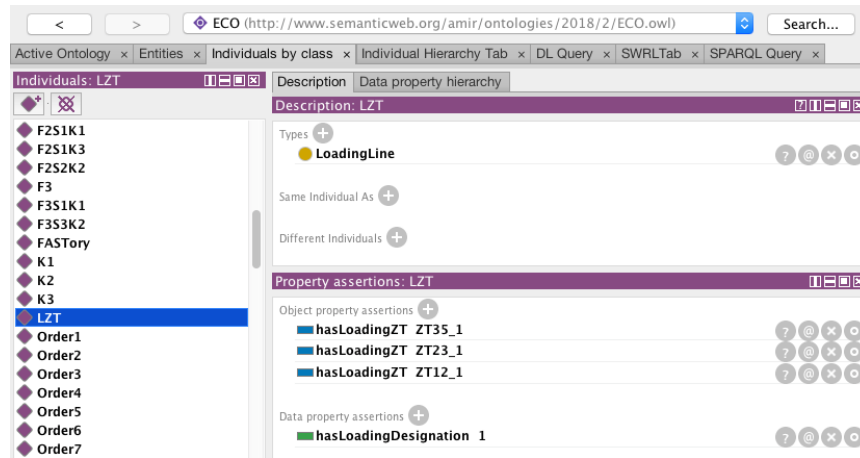
Subsequently, the *hasBypassZT* and *hasBypassDesignation* object and data properties are added as supplementary properties to demonstrate the BZT association. For example, the BZT of work cell five is instantiated as *BZT5* as shown in Figure. 25. This individual is extended with two *hasBypassZT* object property assertions to individuals *ZT14\_5* and *ZT45\_5*, and bypass designation “5”.



**Figure. 25:** Bypass zone transfer 5

And finally, the *hasLoadingZT* and *hasLoadingDesignation* object and data properties are added as supplementary properties to demonstrate the LZT association. The LZT of work cell one is instantiated as *LZT* and it is extended with three *hasLoadingZT* object

properties with individuals *ZT12\_1*, *ZT23\_1*, and *ZT35\_1*, and loading designation “1”, as shown in Figure. 26.

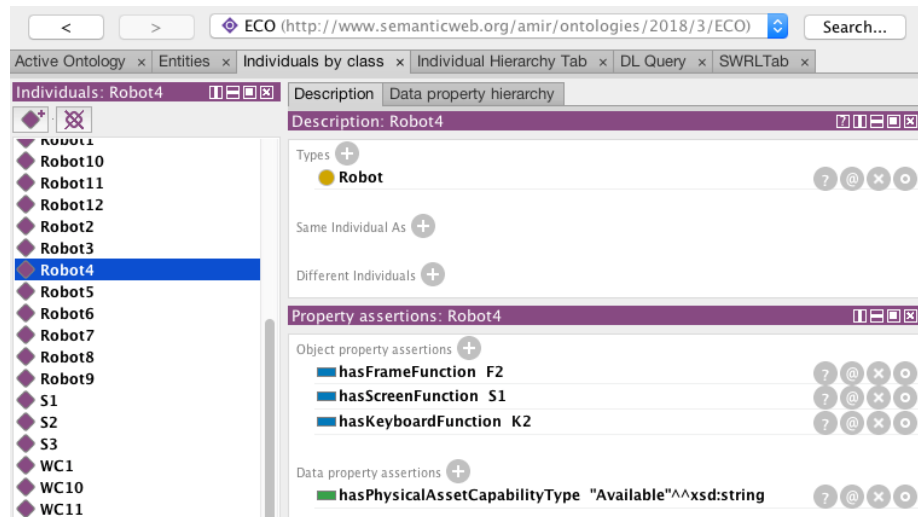


**Figure. 26:** Loading zone transfer in *WC1*

Some of the other FASTory specific extensions to ECO will be explained in the next sections while presenting the semantic rules.

#### 4.2.5 Fixed FASTory layout

In the standard configuration of FASTory, all robots are capable of performing every single assembly operation, given that they have an operational (available) status. The following section details certain modifications performed on the FASTory layout to properly demonstrate some of the semantic rules, which will be explained in the next subsections. To do so, the functionalities of each robot are limited to three assembly operation (frame, screen, and keyboard) with specific patterns, while the use of different pens has not been considered. To model this, additional supporting object properties have been added to the ECO mode. The *hasFunction* object property has been added to denote a robot’s assembly capabilities and is further extended with the *hasFrameFunction*, *hasScreenFunction*, and *hasKeyboardFunction* sub-properties with the range of *DrawFrame*, *DrawScreen*, and *DrawKeyboard*, respectively. For instance, *Robot4* located in *WC4*, has the capability to perform *F2*, *S1*, and *K2* assembly operations as shown in Figure. 27.



**Figure. 27:** Modified functionality of robot 4

The same method has been applied for all robots and the updated configuration of FASTory is shown in Table. 7. According to the updated layout, every single assembly operation can be found in either three or four work cells. For instance, the assembly operation F3, can be performed in WC2, WC3, and WC6, while the operation S3 can be performed in WC5, WC8, WC9, and WC10.

**Table. 7:** Updated FASTory configuration

Work cell name	Assigned functionality
WC2	F3, S1, K3
WC3	F3, S2, K3
WC4	F2, S1, K2
WC5	F2, S3, K1
WC6	F3, S1, K1
WC8	F1, S3, K1
WC9	F1, S3, K2
WC10	F1, S3, K1
WC11	F2, S2, K3
WC12	F1, S2, K2

### 4.3 Semantic rules

The semantic rules added to the model aim to increase the expressivity of the model by linking some of the underlying concepts and relationships, and additionally checking specific requirements for performing use case related operations. The rules are presented as part of two subsections, namely “operational” and “product requirement” rules.

### 4.3.1 Operational SWRL rules

Table. 8 describes the notation of the operational SWRL rules implemented in the ECO model. The table is followed by a detailed explanation of every rule, in addition to some of the concepts considered in the creation of the rules, and the specific object properties added to ECO for supporting the rules. The inferences of the rules are shown in chapter 5.

**Table. 8:** Operational SWRL rules notation

No.	Rule notation
S1	WorkCenter(?x)^WorkUnit(?y)->contains(?x,?y)
S2	Equipment(?x)^hasPhysicalAssetID(?x,?y)->AssetHierarchy(?x)
S3	JobOrder(?x)^requiresPhysicalAsset(?x,?y)^hasPhysicalAssetCapabilityType(?y, "Available")^Pallet(?m)^hasPhysicalAssetCapabilityType(?m,"Available") ->assignedTo(?x, ?y)^assignedTo(?x, ?m)
S4	JobOrder(?x)^assignedTo(?x,?y)^assignedTo(?x,?m)^hasPriority(?x,?p)^ swrlb:greaterThan(?p, 2) ->hasJobOrderStatus(?x,"Ready")
S5	JobOrder(?x)^referencesWorkMaster(?x,?y)^WorkDirective(?p)^hasReferenceTo(?p, ?y)->assignedToJobOrder(?p, ?x)
S6	Discrete(?x)^Robot(?y)^hasFunction(?y,?x)-> requiresPhysicalAsset(?x,?y)

The S1 rule enables the role-based assignment of assets. The successful activation of this rule should automatically assign all the instances of work units to a work center, removing the need for manual addition of the *contains* object property to the work center instance, hence saving time in the modelling phase.

The S2 rule, will automatically associate the assets of an enterprise that have a physical asset ID as, also as instances of the type asset hierarchy. This rule is used as a means of connecting the underlying hierarchy and resource sub-ontologies within the ECO model.

The detailed scheduling activity of the production operation management, receives work models from other activities in the MOM to create work schedules. These work schedules contain the collection of job orders for execution and their sequencing and use job orders to associate the physical processes to particular equipment for production operations. Additionally, job orders reference work masters. The S3 rule, details the assignment of job orders to specific resources. This is done first by checking that the required physical asset (i.e., robots) for a job order has capability type “Available”, and second finding a pallet instance that also holds the same capability type “Available”. With the

conjunction of all the predicates set to true, the job order should be assigned to the robot and the pallet. The *assignedTo* supplementary object property has been added to the ECO model for this resource assignments.

The S4 rule states that if a job order has a priority level higher than “2”, the status of job order would become “Available”, meaning that the job order is ready for execution, assuming other requirements have been met. In order to do that, a complementary object property, called *hasJobOrderStatus*, has been added to ECO to enable the status update of the job order. Additionally, the rule uses the comparison built-ins of SWRL that provide the necessary extra expressivity for this case. The use of priority level for job orders issued from scheduling function, is beneficial to compare the precedence of job orders for execution. The consequent request for the execution of the job order, is sent to the shop floor, i.e., levels one and two of the functional hierarchy. The details concerning these levels is both, beyond the extent of ISA-95 and this implementation.

As presented in section 3.1.3, work masters and work directive collectively define the work definition in MOM. Work directives are copies of work masters that correspond to job orders and have the necessary information for performing them. Every work directive is assigned to a job order. It was also established that job orders are defined by work masters. The S5 rule, enables the mentioned assignment by stating that if a job order and work directive both reference the same work master, the work directive should be assigned to the job order. The *hasReferenceTo* and *assignedToJobOrder* object properties, have been added to ECO to enable this rule to function.

Process segments have been defined in the resource class and identify the various resources required for any production segment. The identification of the required resources may depend on the functionality that they provide. Discrete process segments were added to the model to represent the different assembly operations available in the line such as *F2* or *S1*. Considering all the available process segments, the S6 rule states that if any robot can provide the functionality required for a process segment, it will be linked with the process segment using the *requiresPhysicalAsset* object property.

### 4.3.2 Product requirement SWRL rules

The following section defines some of the rules used for identifying the necessary operations needed for order entries submitted to the FASTory line. On this account, the main goal is to only show the necessary processes as a collection of steps to be used by the associated functional groups to start the operation, perform assembly and zone transfer operations, and finally complete the order. The collection can be seen as a blueprint of steps required to complete an order entry. Hence the actual execution of operations (e.g., assembly and zone transfer) is beyond the scope of this work. Table. 9 details the notation of some of the product requirement SWRL rules used for achieving the identification of product requirements. The notations are subsequently explained in more de-

tails in addition to the concepts considered for the creation of the rules, as well as specific object properties added to supplement the process. The inferences of the rules are shown in chapter 5.

In order to better understand the following section, it should be noted that designations assigned to any individual, play a key role in a large number of rules and logics presented here in after. This has been considered throughout the previous implementation sections as well. For instance, in the case of robot five, the designation “5” has been assigned in the following assertions (using the Integer data type), shown according to their axioms:

- *Robot5 hasDesignation 5*
- *AZT5 hasAssemblyZT 5*
- *BZT5 hasBypassZT 5*

Hence, assigning the same designation to the elements within a work cell (e.g., robots, assembly and bypass zone-transfers), enables the utilization of the SWRL comparison built-ins that help in enabling the goal of the rules.

**Table. 9:** Product requirement SWRL rules

No.	Rule notation
S7	Frame(?x)^hasFrameFunction(?q,?x)^ Screen(?y)^hasScreenFunction(?w,?y)^ Keyboard(?z)^hasKeyboardFunction(?e,?z) -> ScreenMadeAt(?y,?w)^KeyboardMadeAt(?z,?e) ^ FrameMadeAt(?x,?q)
S8	OrderRequests(?m)^Frame2(?x)^hasFrameRequest(?m,?x)^ FrameMadeAt(?x,?y)^hasDesignation(?y,?p)^ FrameMadeAt(?x,?t)^hasDesignation(?t,?o)^ FrameMadeAt(?x,?l)^hasDesignation(?l,?b)^ swrlb:lessThan(?p,?o) swrlb:lessThan(?o,?b) hasPhysicalAssetCapabilityType(?y "Available") -> FrameTo(?m, ?y)
S9	OrderRequests(?m)^Screen1(?x)^hasScreenRequest(?m,?x) ^ ScreenMadeAt(?x,?y)^hasDesignation(?y?p) ScreenMadeAt(?x,?t)^hasDesignation(?t,?o) ScreenMadeAt(?x,?l)^ hasDesignation(?l,?b) swrlb:lessThan(?p,?o) ^ swrlb:lessThan(?o,?b) ^ hasPhysicalAssetCapabilityType(?y,"Available") ->ScreenTo(?m,?y)
S10	OrderRequests(?m)^Keyboard3(?x)^hasKeyboardRequest(?m,?x)^ KeyboardMadeAt(?x,?y)^hasDesignation(?y,?p)^ KeyboardMadeAt(?x,?t)^hasDesignation(?t,?o)^ KeyboardMadeAt(?x,?l)^hasDesignation(?l,?b)^ swrlb:lessThan(?p,?o)^swrlb:lessThan(?o,?b)^ hasPhysicalAssetCapabilityType(?y,"Available") -> KeyboardTo(?m,?y)



S11	OrderRequests(?x)^ hasAssemblyDesignation(?x,?y)^ AssemblyLine(?t)^hasAssemblyDesignation(?t,?y)^hasAssemblyZT(?t,?p) ->hasAssemblyZT(?x,?p)
S12	OrderRequests(?x)^hasBypassDesignation(?x,?y)^ Bypass(?t)^hasBypassDesignation(?t,?y)^hasBypassZT(?t,?p) ->hasBypassZT(?x,?p)
S13	OrderRequests(?x)^hasLoadingDesignation(?x,?y) LoadingLine(?t)^hasLoadingDesignation(?t, ?y)^hasLoadingZT(?t,?p) -> hasLoadingZT(?x,?p)
S14	OrderRequests(?x)^hasFrameRequest(?x,?q)^ hasScreenRequest(?x,?w)^hasKeyboardRequest(?x,?e) -> hasFrameOperation(?x,?q)^hasScreenOperation(?x,?w)^ hasKeyboardOperation(?x,?e)
S15	OrderRequests(?x)^ FrameTo(?x,?q)^hasDesignation(?q,?r)^ ScreenTo(?x,?w)^hasDesignation(?w,?t)^ KeyboardTo(?x,?e)^hasDesignation(?e,?y)^ Bypass(?a)^hasBypassDesignation(?a,?f)^ swrlb:greaterThan(?f,1)^swrlb:lessThan(?f,13)^ swrlb:notEqual(?f,?r)^swrlb:notEqual(?f,?t)^swrlb:notEqual(?f,?y)^ swrlb:notEqual(?t,?y)^swrlb:notEqual(?y,?r)^swrlb:notEqual(?t,?r) -> hasAssemblyDesignation(?x,?r)^hasAssemblyDesignation(?x,?y)^ hasAssemblyDesignation(?x,?t)^hasBypassDesignation(?x,?f)^ hasLoadingDesignation(?x,1)

The S7 rule, details the equipment that are technically capable of performing the assembly operations required by any of the process segments in the line. It should be noted that the S7 rule is based on a modification of the previous rule (S6) to allow achieving the final objective of product requirement identification. Successful activation of this rule, assigns the equipment that have the technical capability of performing a process segments operational needs, to the process segment. This association is done with three new supplementary object properties as *FrameMadeAt*, *ScreenMadeAt*, and *Keyboard-MadeAt*, each associating every robot to the corresponding process segment.

As it was previously established, each robot's functionality (process segments they are capable of performing) was limited to three operations in the fixed FASTory layout. On one hand, according to the fixed layout, every single assembly operation can be found in either three or four work cells. The previous rule helps in identifying all robot instances capable of performing those assembly operations. On the other hand, each order entry has three assembly requests, modelled by associating the requests to process segments, hence each request can technically also be performed in three or four robots. With the options available, in order to choose which robot should perform an assembly request, a specific selection logic was used. The logic uses the designation of the robots to identify their position in FASTory line. For instance, robot 2 located in WC2 is positioned before robot 3 located in WC3, as seen in Figure. 20. Based on the logic, every

assembly request of an order will be send to the *first* work cell containing the robot capable of performing that assembly request.

In order to model this behavior, every robot has been identified with a designation corresponding to their name, using the supplementary *hasDesignation* data property. The S7 rule will additionally add the *FrameTo*, *ScreenTo*, and *KeyboardTo* object properties (based on inferred knowledge) to the order and denote at which robot the assembly request should be processed. The S8 rule, details the logic used for process segment F2 and states that if the F2 operation can be performed at three different robots, the robot with the lowest designation number, should be selected. Subsequently, the rule will only function if the selected robot holds an “Available” status, which should result in the selection of the appropriate robot for F2, based on the logic. The same approach and logic have been used for all other process segments as well. For instance, rules S9 and S10 demonstrate the logic process for the S1 and K3 operations. To keep the section concise, the rules for other process segments are not further demonstrated. With the help of previously typed rules (such as S8, S9 and S10), every order entry will have the *FrameTo*, *ScreenTo*, and *KeyboardTo* object properties, identifying the first robots in the line capable of performing those requests. It should also be noted that the assembly requests of an order can be performed at three, two, or just one distinct robots, depending on whether a robot is capable of performing more than one of the assembly requests. For example, based on the fixed FASTory layout, robot 3 is capable of performing F3, S2, and K3 operations, hence an order entry containing similar assembly requests can perform its assembly operations completely at robot 3.

The S11 rule states that if an order entry and a work master of type *AssemblyLine* have the same assembly designation, the assembly zone-transfers (AZT) of the work master should be copied to the order. The same approach has been used for rules S12 and S13, which copy the bypass zone-transfers (BZT) and loading zone-transfers (LZT) of the *Bypass* and *LoadingLine* work masters, respectively. The S14 rule, transforms the assembly requests into assembly operations to be performed at their corresponding robots, using three supplementary properties as *hasFrameOperation*, *hasScreenOperation*, and *hasKeyboardOperation*. The collection of the S11-14 rules are used for the final product requirement identification that is explained in the next step.

The S15 rule, defines the complete steps required for the completion of an order entry and assigns them (as inferred knowledge) to the order. The knowledge will be added to the order to provide a blueprint for the necessary operations for an order since its entry into WC1, assembly operations and zone-transfer routings in WC2-6 and WC8-12, and routing through WC7. In the rules notation, every robot associated with order requests is identified with its designation. The successful activation of the rule should add the following data properties to the order:

- The designations of the assembly robots using the *hasAssemblyDesignation* property
- The designation of every bypass work master not equal to the designation of assembly robots using the *hasBypassDesignation*
- The *hasLoadingDesignation*, having a value of “1”

Hence, the order will have assembly, bypass, and loading designations, which will subsequently add the necessary routing required by copying the AZT, BZT, and LZT of the designations, through the S11, S12, and S13 rules. At this stage, the order entry should show the following inferred object properties:

- The robots associated to the *FrameTo*, *ScreenTo*, and *KeyboardTo* properties
- The operations required in form of *hasFrameOperation*, *hasScreenOperation*, and *hasKeyboardOperation*
- The necessary routing required in using the *hasAssemblyZT*, *hasBypassZT*, and *hasLoadingZT*

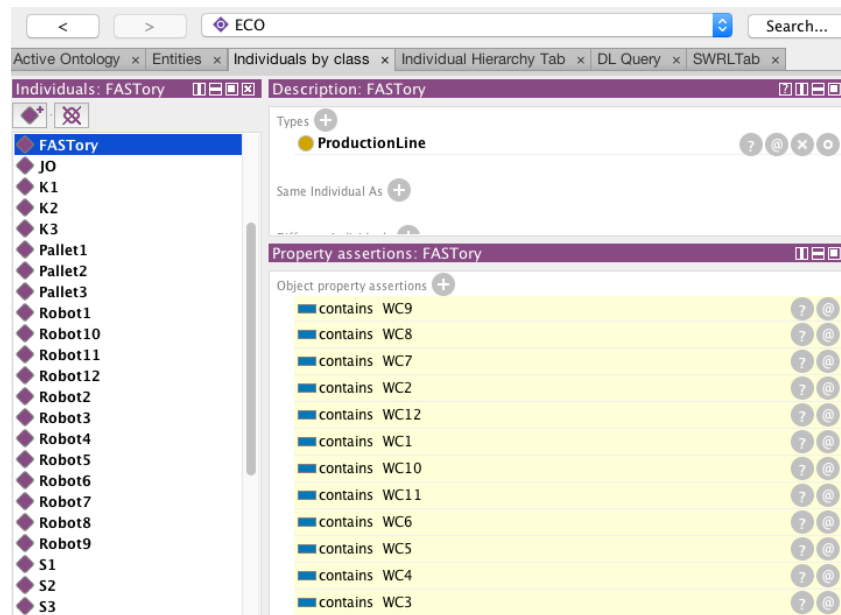
The S15 rule is specifically defined for orders made up of assembly requests that should be performed at three distinct robots. Hence, there are more rules that employ the same logic and demonstrate other scenarios, however, as with the case of rules S8-10, they are not detailed to keep the section concise.

## 5. RESULTS AND DISCUSSION

This section details the results of the implementations explained in the previous chapter. The results are demonstrated as inferred knowledge of the individuals. It should be noted that the inferred knowledge is only shown when the reasoner of the ontology editor is active. The reasoner selected for the knowledge inference is the “Pellet” reasoner, one of the built-in reasoners of Protégé.

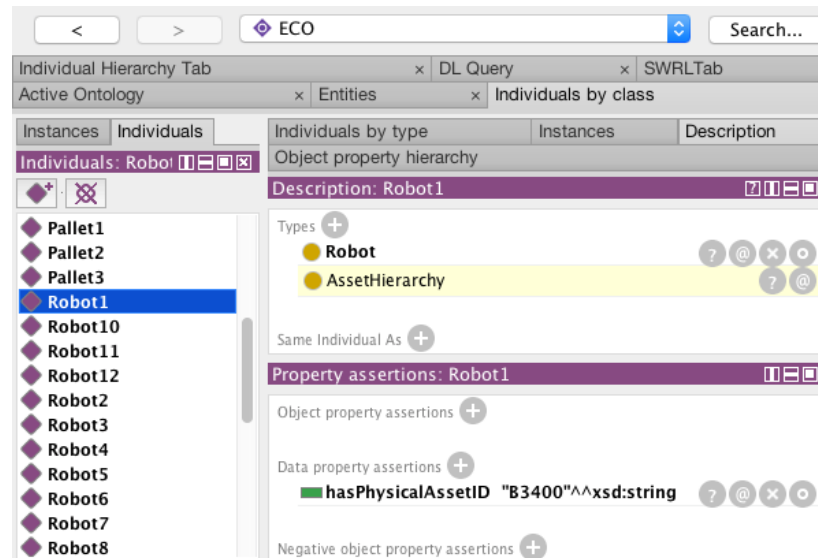
### 5.1 Operational case

The S1 rule detailed the assignment of lower-level elements of the role-based hierarchy to the higher-level elements. By successful triggering of S1, the individual *FASTory* as a type of *ProductionLine* contains all the instances of *WorkCell* available, i.e., all the individuals such as *WC1*, *WC2*, and etc., in the implementation. With the reasoner activated, the inferred knowledge (shown in yellow) is demonstrated as in Figure. 28.



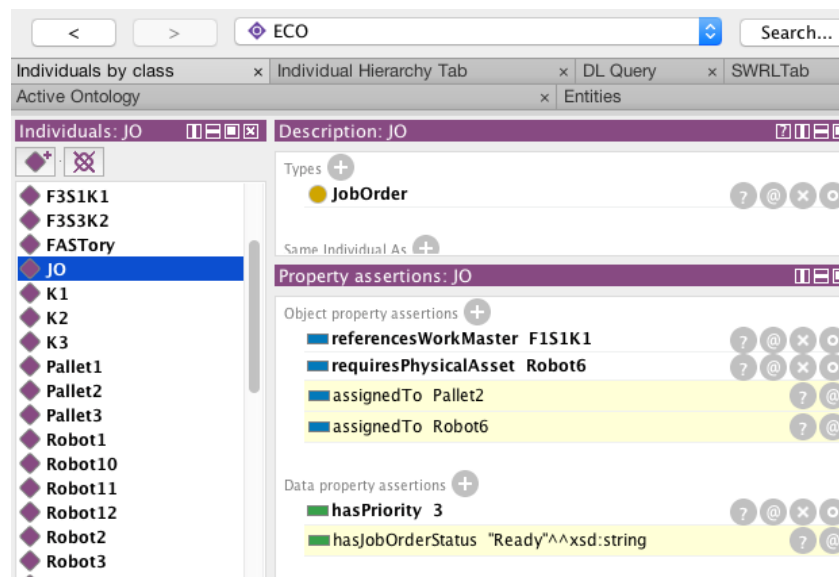
*Figure. 28: FASTory instance inferences*

In the implementation, the physical assets of FASTory were created as type of *Equipment*, subclass of *Resource*. The data property *hasPhysicalAssetID* has been added to specific robots. By the triggering of the S2 rule and with the reasoner active, the robot individuals such as *Robot1* that have the physical asset ID, are added also as a type of *AssetHierarchy*, as shown in Figure. 29.



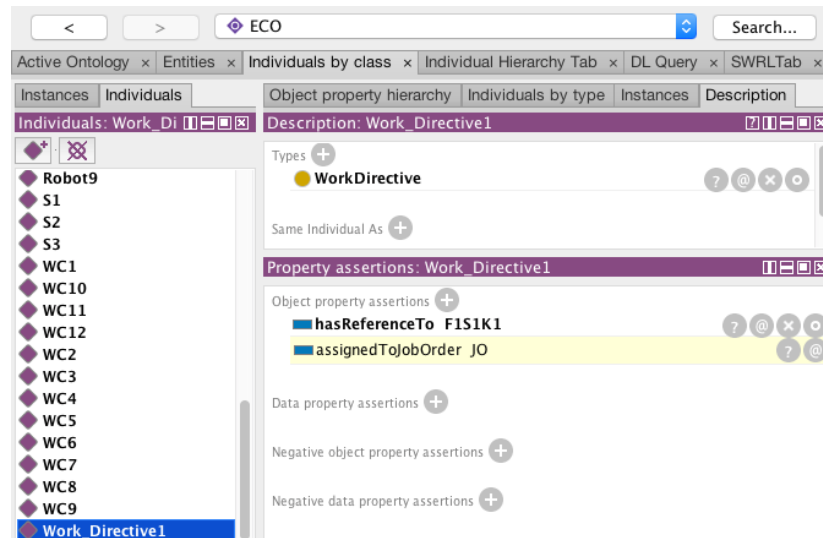
**Figure. 29: Robot1 instance inference**

The triggering of the S3 rule, assigns *Robot6* and *Pallet2* to the job order *JO* as shown in Figure. 30. In this case, both of the equipment had an “Available” status. Additionally, the job order *JO*, has a priority status “3”. Hence, the S4 rule is triggered, adding the order status of job order to “Ready” (as inferred knowledge), making the job order set for execution, which is also shown in Figure. 30.



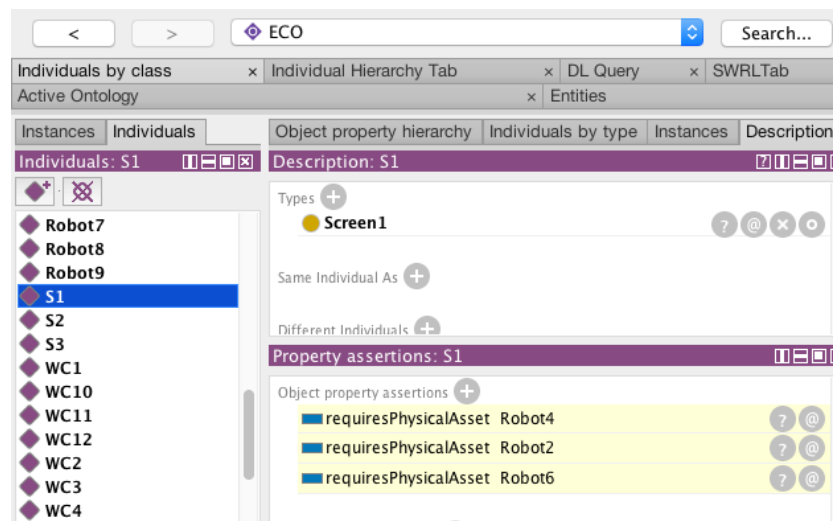
**Figure. 30: Job order inferences**

The instance *Work\_Directive1* references the instance *F1S1K1*, which is a work master of type *ProductDefinition* as previously explained. As the job order *JO* also references the same work master as seen in Figure. 30, the work directive is assigned to job order individual *JO* as shown in Figure. 31.



**Figure. 31:** Work directive inference

And finally, the triggering of the S6 rule results in the identification of every resource capable of performing the process segments. For instance, the inferred knowledge for process segment *S1* is shown in Figure. 32. The identification of the required equipment has a direct impact on whether a job order will be assigned to them or not.



**Figure. 32:** Process segment inference

## 5.2 Product requirements case

The following subsection details the identification of necessary requirements for order entries to the FASTory line, referred to as product requirements. In order to demonstrate this, each order entry is first explained based on their assembly requests, followed by the availability status of every robot at the moment of order entry. Subsequently, the product requirement results for the order are presented as inferred knowledge of the individuals.

The inferred (new) knowledge can also be shown using SPARQL queries, in the same inferred state, or as explicit knowledge when the users transfer the inferred knowledge to the model. Hence, in this case, in order to demonstrate a user's capability to query the knowledge they require, some illustrative queries for zone-transfers are shown using the interface of the "Snap SPARQL query" in Protégé, which enables the query of inferred knowledge, with an active reasoner. The first order, i.e., *Order1*, consists of the requests F3, S1, and K3. The status of every robot in FASTory was set to "Available" for this order. The product requirements for the order are shown in Figure. 33.

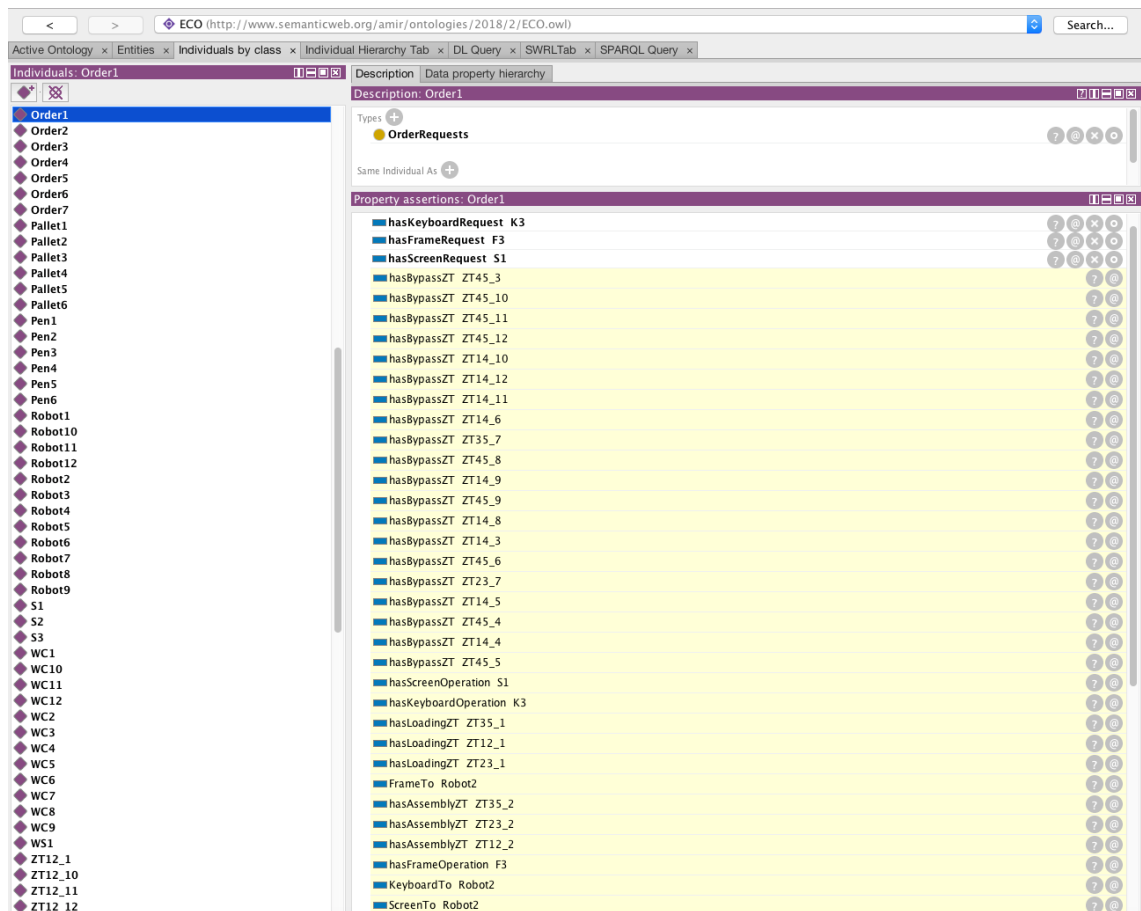
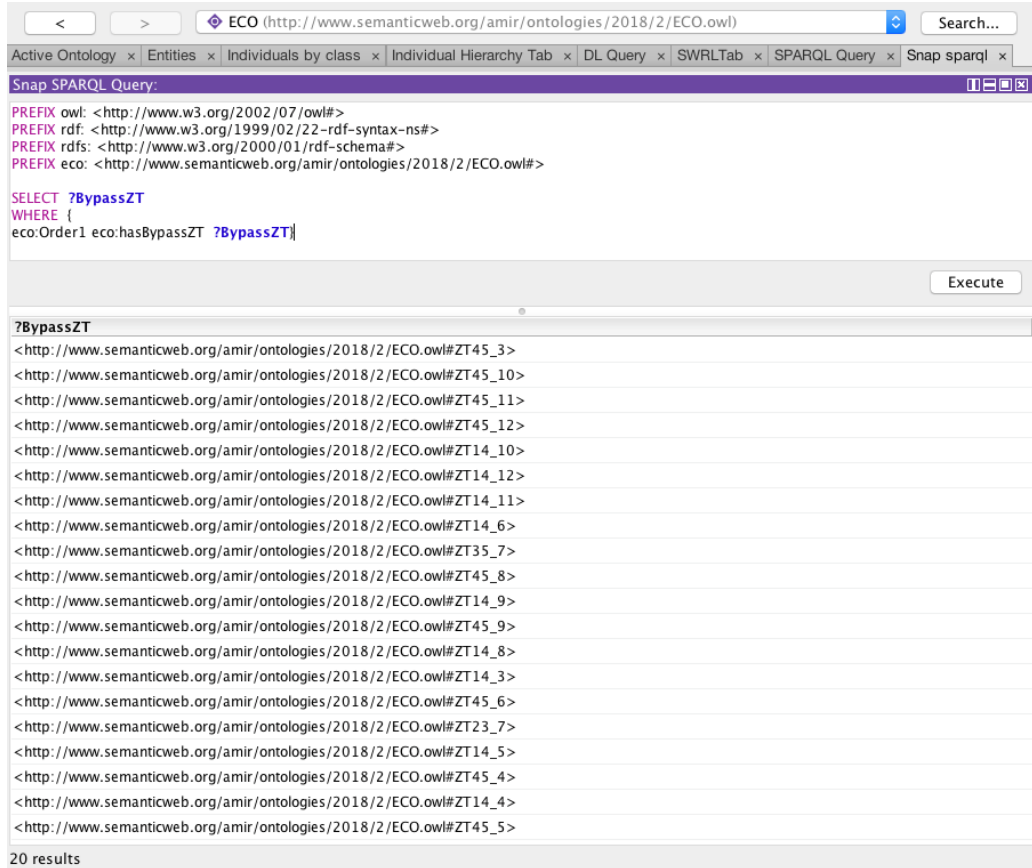


Figure. 33: *Order1* product requirements

Based on the fixed FASTory layout and the available status of every robot, the product requirements are shown correctly as inferences and it can be seen that all of the *Order1* assembly requests are identified to be performed at *Robot2* that is capable of performing all of the necessary requests. Additionally, in the case of *Order1*, the bypass zone-transfers can be queried and shown as in Figure. 34. The results of the query match the inferred bypass zone-transfers shown in Figure. 33.



**Figure. 34:** Query of bypass zone-transfers of *Order1*

In case of *Order2*, the order consists of F1, S1, and K2. The statuses of *Robot8*, *Robot9*, and *Robot10* were set as “Not-Available” while the other robots were set as “Available”. The product requirements for the order are shown in Figure. 35. Based on the selection logic explained in section 4.3.2, the F1 frame request can be performed at robot 8, 9, 10, or 12, and should be sent to the first robot capable of that request but since *Robot8*, *Robot9*, and *Robot10* were set as “Not-Available”, the frame request is sent to *Robot12*, the only robot with holds a status “Available”. Additionally, the inferences show that the order should be transferred through the bypass conveyor path in robots 8-10. The assembly zone-transfers of *Order2* can be queried and shown as in Figure. 36. The query results show a total of nice results, i.e., the three zone-transfer operations needed at robots 2, 4, and 12.



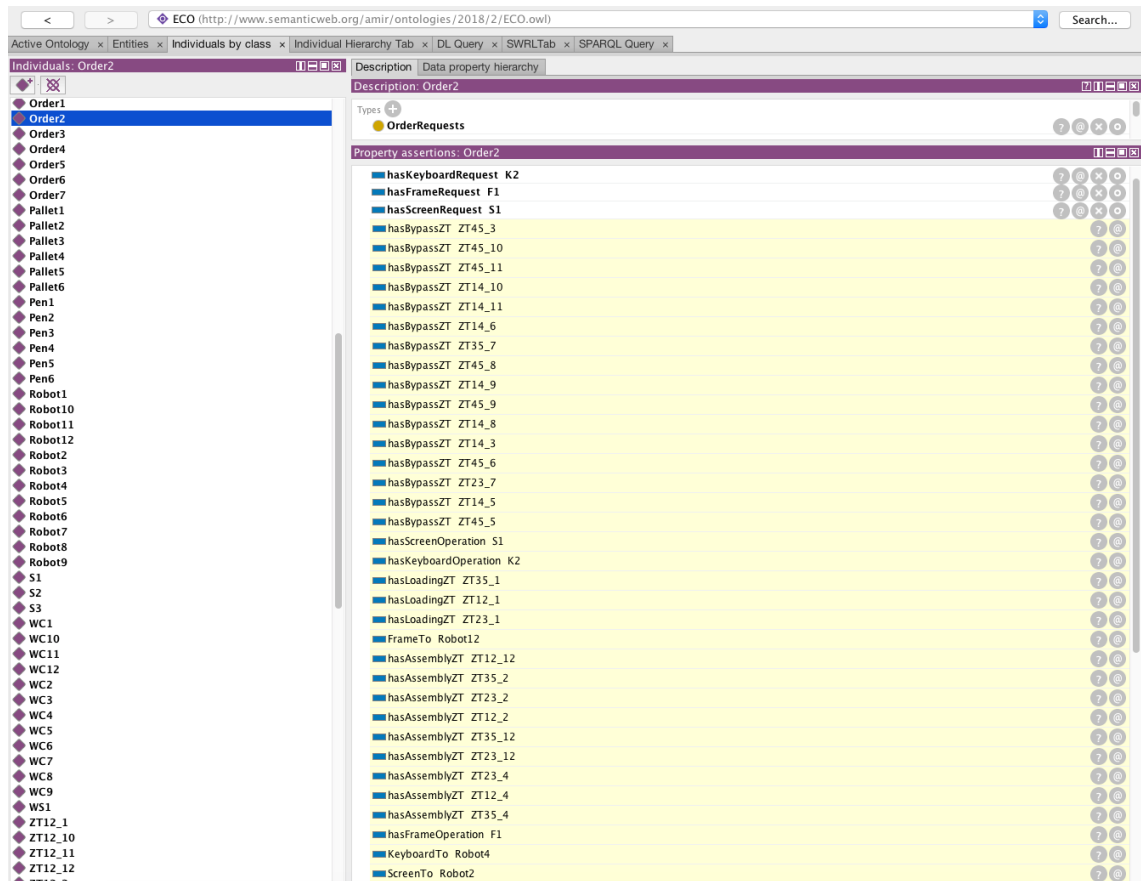


Figure. 35: Order2 product requirements

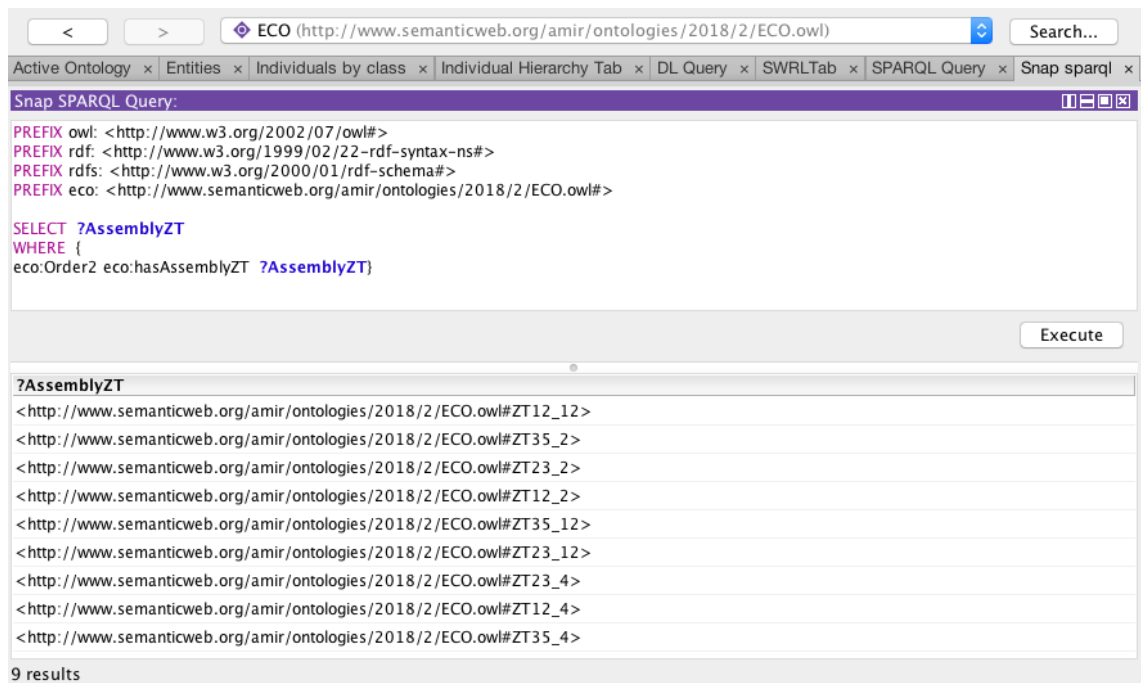


Figure. 36: Query of assembly zone-transfers of Order2

The next order, i.e., *Order3*, consists of F2, S3, and K2, assembly requests. The status of the robots remained the same as in the last order, except *Robot4* that also held a “Not-Available” status, as did *Robot8*, *Robot9*, and *Robot10*.

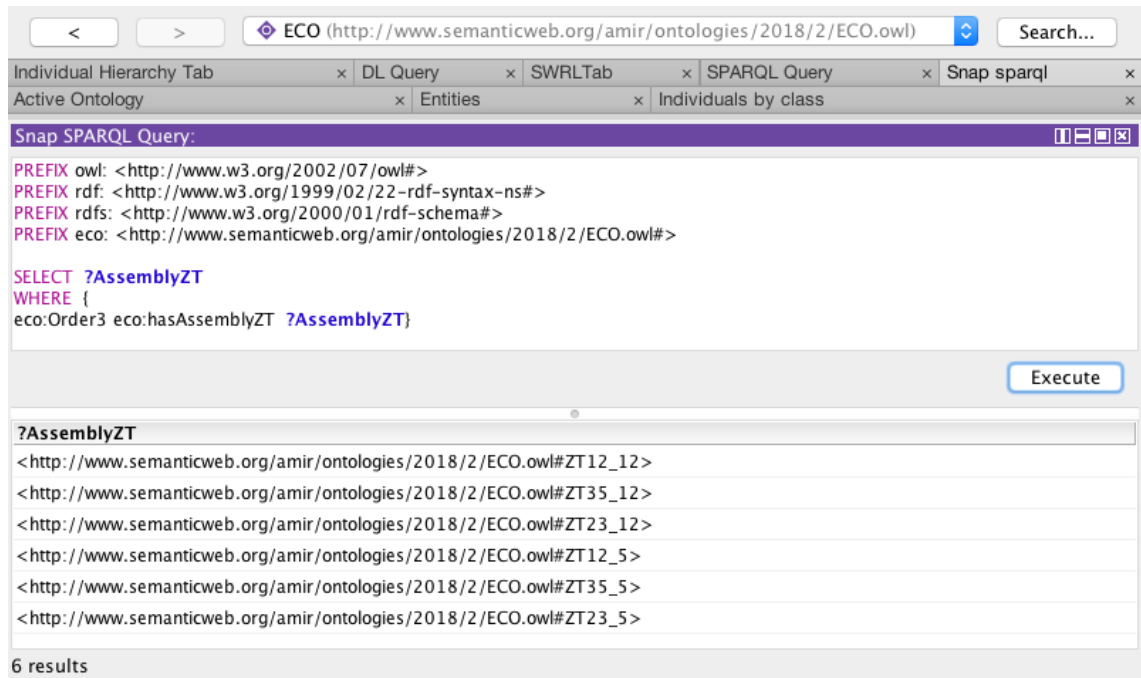
In the fixed FASTory layout, *Robot4* has the functionalities to perform the F2 and K2 assembly requests and it is also the first robot in the layout as per the selection logic. But because *Robot4* holds a “Not-Available” status, F2 assembly will be identified to be carried out at *Robot5* (the next station with status “Available”), while K2 assembly will be identified to be carried out at *Robot12*, the next station with status “Available” as others stations capable of performing K2 are “Not-Available”. Hence, the product requirements of *Order3* are shown accordingly in Figure. 37.

The screenshot shows a web application interface for an ontology. The left sidebar displays a tree view of entities, with *Order3* selected. The main panel shows the description of *Order3* and a table of property assertions. The table lists various assembly requests and their corresponding robot assignments.

Property	Value
hasScreenRequest	S3
hasKeyboardRequest	K2
hasFrameRequest	F2
hasBypassZT	ZT45_2
hasBypassZT	ZT45_3
hasBypassZT	ZT45_10
hasBypassZT	ZT45_11
hasBypassZT	ZT14_10
hasBypassZT	ZT14_11
hasBypassZT	ZT14_6
hasBypassZT	ZT35_7
hasBypassZT	ZT45_8
hasBypassZT	ZT14_9
hasBypassZT	ZT45_9
hasBypassZT	ZT14_8
hasBypassZT	ZT14_3
hasBypassZT	ZT45_6
hasBypassZT	ZT14_2
hasBypassZT	ZT23_7
hasBypassZT	ZT45_4
hasBypassZT	ZT14_4
hasScreenOperation	S3
hasKeyboardOperation	K2
hasLoadingZT	ZT35_1
hasLoadingZT	ZT12_1
hasLoadingZT	ZT23_1
FrameTo	Robot5
hasAssemblyZT	ZT12_12
hasAssemblyZT	ZT35_12
hasAssemblyZT	ZT23_12
hasAssemblyZT	ZT12_5
hasAssemblyZT	ZT35_5
hasAssemblyZT	ZT23_5
hasFrameOperation	F2
KeyboardTo	Robot12
ScreenTo	Robot5

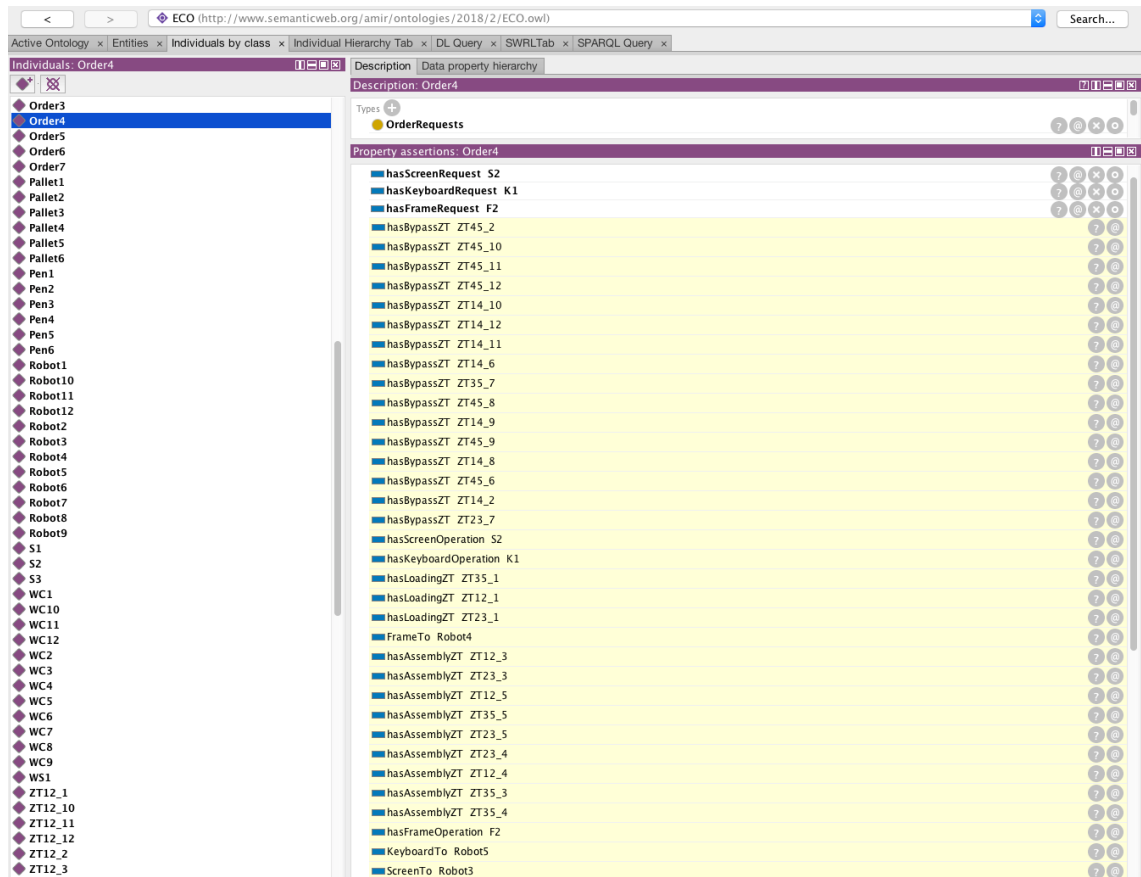
**Figure. 37:** *Order3* product requirements

As it can be seen from the inference, the S3 assembly request is also identified to be carried out at *Robot5*. This is further illustrated in the query of assembly zone-transfers of *Order3*, as shown in Figure. 38.



**Figure. 38:** Query of assembly zone-transfers of Order3

The next order, i.e., *Order4*, consists of the assembly requests F2, S2, and K1. The status of all the robots in FASTory was set to “Available”. As per the selection logic and the available status of all robots, it can be seen in Figure. 39 that the product requirements are clearly identified to be carried out at *Robot3*, *Robot4*, and *Robot5*, hence requiring the transfer through the assembly conveyor path of work cells 3-5 while using the bypass conveyor path for the remaining work cells.



**Figure. 39:** Order4 product requirements

The last order entry shown, i.e., *Order5*, consists of the assembly requests F3, S3, and K3. The statuses of robots 2-5 were set to “Not-Available” while robots 6-12 were “Available”. Based on the fixed layout, robot statuses, and the order assembly requests, the necessary assembly and transfer operations are identified and shown in Figure. 40. As it can be seen, the main assembly operations are identified to be performed at *Robot6*, *Robot8*, and *Robot11*. Hence, the order needs to be transferred through the bypass zone- transfers of work cells 2-5, before starting the F3 assembly operation at *Robot6*.

Active Ontology: ECO (http://www.semanticweb.org/amir/ontologies/2018/2/ECO.owl)

Individuals: Order5

Description: Order5

Types: OrderRequests

Property assertions: Order5

hasFrameRequest	F3
hasScreenRequest	S3
hasKeyboardRequest	K3
hasBypassZT	ZT45_2
hasBypassZT	ZT45_3
hasBypassZT	ZT45_10
hasBypassZT	ZT45_12
hasBypassZT	ZT14_10
hasBypassZT	ZT14_12
hasBypassZT	ZT35_7
hasBypassZT	ZT14_9
hasBypassZT	ZT45_9
hasBypassZT	ZT14_3
hasBypassZT	ZT14_2
hasBypassZT	ZT23_7
hasBypassZT	ZT14_5
hasBypassZT	ZT45_4
hasBypassZT	ZT14_4
hasBypassZT	ZT45_5
hasScreenOperation	S3
hasKeyboardOperation	K3
hasLoadingZT	ZT35_1
hasLoadingZT	ZT12_1
hasLoadingZT	ZT23_1
FrameTo	Robot6
hasAssemblyZT	ZT12_11
hasAssemblyZT	ZT35_11
hasAssemblyZT	ZT23_11
hasAssemblyZT	ZT12_8
hasAssemblyZT	ZT23_8
hasAssemblyZT	ZT35_8
hasAssemblyZT	ZT35_6
hasAssemblyZT	ZT12_6
hasAssemblyZT	ZT23_6
hasFrameOperation	F3
KeyboardTo	Robot11
ScreenTo	Robot8

*Figure. 40: Order5 product requirements*

## 6. CONCLUSION AND FUTURE WORK

### 6.1 Conclusion

The shift towards a more networked world, brings along an abundance of data that becomes available detailing every minor and significant event in the world. In the context of modern and smart manufacturing systems, the need to handle this data in form of information exchanges between functional groups of a system, is based on a clear comprehension of the system at hand, and a method for managing the data and handling the information exchange. This thesis proposes an approach for designing a manufacturing systems model, identified in this thesis work as ECO, using a KR&R formalism, i.e., an ontology, based on the ISA-95 standard.

The decision to model the manufacturing domain using an ontology has been made because it enables modelling the entities and relationships in the domain with the desired expressivity, while allowing an easy method to add new knowledge to the KB, hence addressing the issue of re-configurability in manufacturing systems. On the other hand, the ISA-95 standard has been referenced since it facilitates the integration of enterprise and control systems, by identifying main elements of a system in different levels of the hierarchy, and their information exchanges. Therefore, modelling the proposed manufacturing system based on the standard, enabled the conformance to a uniform set of concepts and guidelines, providing the basis for a generic manufacturing systems model, that allows easy extendibility.

It has been one of the main goals of this thesis work to mitigate and possibly remove inconsistencies while adopting the standard by following the concepts, guidelines, and taxonomies as much as possible to maintain the generic nature of the model. And as one of the benefits of using ontologies is reuse of existing knowledge, the generic nature of the proposed solution should allow easy reuse of the model in the manufacturing domain. Furthermore, the manufacturing model has been implemented in an industrial use case (i.e., the FASTory) to demonstrate the applicability of the solution in the domain of manufacturing. The FASTory implementation highlights the extendibility and re-configurability of the ECO model with the addition of FASTory specific resources and operations to the generic model.

The main contribution of this work is the addition of semantic rules, i.e., SWRL rules, in the model. Semantic rules are added to the ECO model with the purpose of increasing the expressivity of model, leading to greater reasoning capabilities, and ultimately enabling the identification of product requirements. The semantic rules can also be consid-

ered as a method for the knowledge engineers to design and shape the understanding of a reasoner to inference implicit knowledge. In the context of this work, SWRL rules were used for the semantic enrichment of the manufacturing systems model. The approach for defining the basic operation of the rules and abundance of built-ins available, make the modelling of the rules easy, even for less experienced users in the field of KR&R. It was not the objective of this thesis work to highlight just the supporting role of the rules, but to show how the rules can enable functionalities essential to the business and operational values of an enterprise and other systems.

As with any other technology, there are aspects to the application of SWRL rules that need to be taken into consideration while implementing them. One of such aspects relates to the extent of its application (complexity) and the number of rules used. In case of possible incorrect inferences due to the rules, the users need to be able to trace back the wrong inference to the rule that is causing the inference. This task becomes more wearying if the rules are modeled with difficult concepts and predicates that unnecessarily complicate the rule without any benefits. Another aspect is the lack of proper documentation and examples for a large number of SWRL built-ins, leaving a possible user with a restricted selection of proven use cases for the built-ins.

The objectives of this thesis work have been validated by the application of the proposed manufacturing systems model (ECO) in an industrial use case. But as with any other solutions, there are aspects that can be improved, which are discussed in the future work.

## 6.2 Future work

The product requirement SWRL rules explained in section 4.3.2, enable the identification of necessary assembly and zone-transfer operations to complete an order. The successful application of those rules results in an unorganized list of steps detailing every single operation in the assembly line required for the completion of that particular order entry. In order to be able to properly execute this list of requirements using the MES/MOM activities such as scheduling, dispatching, and execution management, the requirements need to be scheduled. Scheduling can be seen as a moderator that allows the proper flow of the operations within the system. A simple example in the context of this work could be the sequencing of zone operations. For instance, when a pallet should transfer from zone 2 to zone 3 within a work cell (ZT23), the operation should only be allowed if the pallet is physically at work zone 2. This in theory can be modelled and asserted in the model by assigning a supplementary data property such as *hasZoneStatus* with data type Boolean to the work zone. Subsequently the desired functionality mentioned can be modeled with a SWRL rule stating that the ZT23 zone-transfer should be performed when work zone 2 has a zone status “True”, denoting that the pallet is physically there. Logically, once the zone-transfer is complete, the status of work zone 2 should become “False”. This last update can also be added using a SWRL

rule, but the “monotonic” nature of SWRL will not allow the change of the already asserted knowledge as explained in section 2.3.7. Hence if the inferred knowledge (updated work zone status) is transferred to the model as asserted knowledge, the work zone will have two similar status properties holding both true and false Boolean data type values. This limitation of SWRL paves the way for the future work of this thesis work.

Proper sequencing requires the availability of large amounts of data from different sources, amongst other requirements. Every event happening in the shop floor will change those data values and modify the status of various elements of the system. These changes and status updates can be inferred using SWRL rules and asserted to the KB, but they cannot be modified. A possible solution for this limitation is the usage of an external engine that allows such modification such as OWLAPI. The OWLAPI can be used to both assert and remove knowledge from the KB. Hence, if the SWRL rules are used in conjunction with OWLAPI, the rules can infer new knowledge to the model, and after asserting the new knowledge into the KB, the OWLAPI is capable of removing the unnecessary knowledge. The mentioned tools can then be used to enable the proper scheduling of operations within FASTory and possibly other use cases.

Another aspect that should be considered for future work with the help of the tools and method mentioned could be the submission of more than one order into the system. The proper assembly and zone-transfer operations for more than one order entry, requires a proper coordination between different equipment and in cases between the orders themselves. Hence, the need for the manipulation of the knowledge in the KB becomes more significant.



## REFERENCES

- [1] “IEEE Standard for System and Software Verification and Validation,” *IEEE Std 1012-2012 Revis. IEEE Std 1012-2004*, pp. 1–223, May 2012.
- [2] “MESA White Paper #58: The Importance of Standards in Smart Manufacturing.” [Online]. Available: <https://services.mesa.org/ResourceLibrary/ShowResource/96cc3f1a-e706-47f1-878f-274840602f41>. [Accessed: 02-Apr-2018].
- [3] “W3C Semantic Web Activity Homepage.” [Online]. Available: <http://www.w3.org/2001/sw/>. [Accessed: 15-Dec-2015].
- [4] R. J. Brachman, H. J. Levesque, and M. Pagnucco, *Knowledge Representation and Reasoning: Knowledge Representation and Reasoning*. San Francisco, UNITED STATES: Elsevier Science & Technology, 2004.
- [5] M. Ahmad, B. R. Ferrer, B. Ahmad, D. Vera, J. L. Martinez Lastra, and R. Harrison, “Knowledge-based PPR modelling for assembly automation,” *CIRP J. Manuf. Sci. Technol.*, Jan. 2018.
- [6] A. Mayr, A. Meyer, P. Gönnheimer, J. Gramlich, M. Reiser, and J. Franke, “Concept for an integrated product and process development of electric drives using a knowledge-based system,” in *2017 7th International Electric Drives Production Conference (EDPC)*, 2017, pp. 1–7.
- [7] *Knowledge Technologies*. Polimetrica s.a.s., 2008.
- [8] H. Panetto and A. Molina, “Enterprise integration and interoperability in manufacturing systems: Trends and issues,” *Comput. Ind.*, vol. 59, no. 7, pp. 641–646, Sep. 2008.
- [9] C. Scheuermann, S. Verclas, and B. Bruegge, “Agile Factory - An Example of an Industry 4.0 Manufacturing Process,” in *2015 IEEE 3rd International Conference on Cyber-Physical Systems, Networks, and Applications*, 2015, pp. 43–47.
- [10] P. Martin and A. D’Acunto, “Design of a production system: An application of integration product-process,” *Int. J. Comput. Integr. Manuf.*, vol. 16, no. 7–8, pp. 509–516, Jan. 2003.
- [11] V. V. Sliusar, O. V. Nikolaev, V. G. Dorogov, L. G. Gagarina, and A. M. Andrianov, “Usage of triggers for business process controlling in ERP systems,” in *2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 2018, pp. 1567–1570.
- [12] M. J. Lee, W. Y. Wong, and M. H. Hoo, “Next era of enterprise resource planning system review on traditional on-premise ERP versus cloud-based ERP: Factors influence decision on migration to cloud-based ERP for Malaysian SMEs/SMIs,” in *2017 IEEE Conference on Systems, Process and Control (ICSPC)*, 2017, pp. 48–53.
- [13] P. Blanc, I. Demongodin, and P. Castagna, “A holonic approach for manufacturing execution system design: An industrial application,” *Eng. Appl. Artif. Intell.*, vol. 21, no. 3, pp. 315–330, Apr. 2008.
- [14] “New ways for the effective factory,” in *Manufacturing Execution Systems — MES*, Springer, Berlin, Heidelberg, 2007, pp. 1–39.
- [15] N. Sachdeva, R. K. Obheroi, A. Srivastava, and S. K. Nehal, “Diffusion of industry 4.0 in manufacturing sector #x2014; An innovative framework,” in *2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS)*, 2017, pp. 1–5.
- [16] K. Zhou, T. Liu, and L. Zhou, “Industry 4.0: Towards future industrial opportunities and challenges,” in *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, 2015, pp. 2147–2152.

- [17] A. Mazak and C. Huemer, "HoVer: A modeling framework for horizontal and vertical integration," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, 2015, pp. 1642–1647.
- [18] V. Krueger *et al.*, "A Vertical and Cyber Physical Integration of Cognitive Robots in Manufacturing," *Proc. IEEE*, vol. 104, no. 5, pp. 1114–1127, May 2016.
- [19] Q. Liu *et al.*, "An Application of Horizontal and Vertical Integration in Cyber-Physical Production Systems," in *2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2015, pp. 110–113.
- [20] R. S. Peres, M. Parreira-Rocha, A. D. Rocha, J. Barbosa, P. Leitão, and J. Barata, "Selection of a data exchange format for industry 4.0 manufacturing systems," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 5723–5728.
- [21] P. Cicconi, A. C. Russo, M. Germani, M. Prist, E. Pallotta, and A. Monteriù, "Cyber-physical system integration for industry 4.0: Modelling and simulation of an induction heating process for aluminium-steel molds in footwear soles manufacturing," in *2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI)*, 2017, pp. 1–6.
- [22] J. Zhang, X. Yao, J. Zhou, J. Jiang, and X. Chen, "Self-Organizing Manufacturing: Current Status and Prospect for Industry 4.0," in *2017 5th International Conference on Enterprise Systems (ES)*, 2017, pp. 319–326.
- [23] "About ISA - The Home for Automation Professionals- ISA." [Online]. Available: <https://www.isa.org/about-isa/>. [Accessed: 05-Apr-2018].
- [24] B. Scholten, *The Road to Integration: A Guide to Applying the ISA-95 Standard in Manufacturing*. ISA, 2007.
- [25] "ANSI/ISA-95.00.01-2010 (IEC 62264-1 Mod) Enterprise-Control System Integration - Part 1: Models and Terminology." [Online]. Available: <https://www.isa.org/store/products/product-detail/?productId=116636>. [Accessed: 03-Apr-2018].
- [26] "ANSI/ISA-95.00.02-2010 (IEC 62264-2 Mod) Enterprise-Control System Integration - Part 2: Object Model Attributes." [Online]. Available: <https://www.isa.org/store/products/product-detail/?productId=116637>. [Accessed: 03-Apr-2018].
- [27] "ANSI/ISA-95.00.03-2013 Enterprise-Control System Integration - Part 3: Activity Models of Manufacturing Operations Management." [Online]. Available: <https://www.isa.org/store/products/product-detail/?productId=116782>. [Accessed: 03-Apr-2018].
- [28] "ANSI/ISA-95.00.04-2012 Enterprise-Control System Integration - Part 4: Objects and attributes for manufacturing operations management integration." [Online]. Available: <https://www.isa.org/store/products/product-detail/?productId=116756>. [Accessed: 03-Apr-2018].
- [29] "ANSI/ISA-95.00.05-2013 Enterprise-Control System Integration - Part 5: Business-to-Manufacturing Transactions." [Online]. Available: <https://www.isa.org/store/products/product-detail/?productId=116783>. [Accessed: 16-May-2018].
- [30] "ANSI/ISA-95.00.06-2014, Enterprise-Control System Integration--Part 6: Messaging Service Model." [Online]. Available: <https://www.isa.org/store/ansi/isa-950006-2014,-enterprise-control-system-integration--part-6-messaging-service-model/38003611>. [Accessed: 16-May-2018].

- [31] “ANSI/ISA-95.00.07-2017, Enterprise-Control System Integration-Part 7: Alias Service Model.” [Online]. Available: <https://www.isa.org/store/ansi/isa-950007-2017,-enterprise-control-system-integration-part-7-alias-service-model/60709545>. [Accessed: 16-May-2018].
- [32] B. Wally, C. Huemer, and A. Mazak, “Entwining plant engineering data and ERP information: Vertical integration with AutomationML and ISA-95,” in *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*, 2017, pp. 356–364.
- [33] B. Wally, C. Huemer, and A. Mazak, “Aligning Business Services with Production Services: The Case of REA and ISA-95,” in *2017 IEEE 10th Conference on Service-Oriented Computing and Applications (SOCA)*, 2017, pp. 9–17.
- [34] H. O. Unver, “An ISA-95-based manufacturing intelligence system in support of lean initiatives,” *Int. J. Adv. Manuf. Technol.*, vol. 65, no. 5–8, pp. 853–866, Mar. 2013.
- [35] L. Prades, F. Romero, A. Estruch, A. García-Dominguez, and J. Serrano, “Defining a Methodology to Design and Implement Business Process Models in BPMN According to the Standard ANSI/ISA-95 in a Manufacturing Enterprise,” *Procedia Eng.*, vol. 63, pp. 115–122, Jan. 2013.
- [36] J. Virta, I. Seilonen, A. Tuomi, and K. Koskinen, “SOA-Based integration for batch process management with OPC UA and ISA-88/95,” in *2010 IEEE 15th Conference on Emerging Technologies Factory Automation (ETFA 2010)*, 2010, pp. 1–8.
- [37] I. M. Delamer and J. L. M. Lastra, “Service-Oriented Architecture for Distributed Publish/Subscribe Middleware in Electronics Production,” *IEEE Trans. Ind. Inform.*, vol. 2, no. 4, pp. 281–294, Nov. 2006.
- [38] S. Staab and R. Studer, *Handbook on Ontologies*. Springer Science & Business Media, 2010.
- [39] E. Shortliffe, *Computer-Based Medical Consultations: MYCIN*. Elsevier, 2012.
- [40] W.-T. Tsai, R. Vishnuvajjala, and D. Zhang, “Verification and validation of knowledge-based systems,” *IEEE Trans. Knowl. Data Eng.*, vol. 11, no. 1, pp. 202–212, Jan. 1999.
- [41] G. L. Rocca, “Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design,” *Adv. Eng. Inform.*, vol. 26, no. 2, pp. 159–179, Apr. 2012.
- [42] “semantics | Definition of semantics in English by Oxford Dictionaries,” *Oxford Dictionaries | English*. [Online]. Available: <https://en.oxforddictionaries.com/definition/semantics>. [Accessed: 25-Sep-2017].
- [43] “Definition of ONTOLOGY.” [Online]. Available: <https://www.merriam-webster.com/dictionary/ontology>. [Accessed: 15-May-2018].
- [44] T. R. Gruber, “Toward principles for the design of ontologies used for knowledge sharing?,” *Int. J. Hum.-Comput. Stud.*, vol. 43, no. 5, pp. 907–928, Nov. 1995.
- [45] M. C. Suárez-Figueroa, A. Gómez-Pérez, E. Motta, and A. Gangemi, *Ontology Engineering in a Networked World*. Springer Science & Business Media, 2012.
- [46] M. Hepp, “Ontologies: State of the Art, Business Potential, and Grand Challenges,” in *Ontology Management*, Springer, Boston, MA, 2008, pp. 3–22.
- [47] “SWRL: A Semantic Web Rule Language Combining OWL and RuleML.” [Online]. Available: <https://www.w3.org/Submission/SWRL/#8>. [Accessed: 27-Apr-2018].

- [48] “Semantic Web - W3C.” [Online]. Available: <https://www.w3.org/standards/semanticweb/>. [Accessed: 26-Sep-2017].
- [49] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen, “From SHIQ and RDF to OWL: the making of a Web Ontology Language,” *Web Semant. Sci. Serv. Agents World Wide Web*, vol. 1, no. 1, pp. 7–26, Dec. 2003.
- [50] “Resource Description Framework (RDF): Concepts and Abstract Syntax.” [Online]. Available: <https://www.w3.org/TR/rdf-concepts/>. [Accessed: 27-Apr-2018].
- [51] “RDF Schema 1.1.” [Online]. Available: <https://www.w3.org/TR/rdf-schema/>. [Accessed: 27-Apr-2018].
- [52] J. M. A. Calero, A. M. Ortega, G. M. Perez, J. A. B. Blaya, and A. F. G. Skarmeta, *© Rinton Press A Non-monotonic Expressiveness Extension on the Semantic Web Rule Language*. .
- [53] J. L. M. Lastra, I. M. Delamer, and F. Ubis, *Domain Ontologies for Reasoning Machines in Factory Automation*. ISA, 2010.
- [54] Y. A. Effendi and R. Sarno, “SWRL rules for identifying short loops in business process ontology model,” in *2017 11th International Conference on Information Communication Technology and System (ICTS)*, 2017, pp. 209–214.
- [55] L. F. Lin, W. Y. Zhang, Y. C. Lou, C. Y. Chu, and M. Cai, “Developing manufacturing ontologies for knowledge reuse in distributed manufacturing environment,” *Int. J. Prod. Res.*, vol. 49, no. 2, pp. 343–359, Jan. 2011.
- [56] M. Cai, W. Y. Zhang, and K. Zhang, “ManuHub: A Semantic Web System for Ontology-Based Service Management in Distributed Manufacturing Environments,” *IEEE Trans. Syst. Man Cybern. - Part Syst. Hum.*, vol. 41, no. 3, pp. 574–582, May 2011.
- [57] Y.-L. Chi, “Rule-based ontological knowledge base for monitoring partners across supply networks,” *Expert Syst. Appl.*, vol. 37, no. 2, pp. 1400–1407, Mar. 2010.
- [58] F. B. Ramis, B. Ahmad, D. Vera, A. Lobov, R. Harrison, and L. J. L. Martínez, “Product, process and resource model coupling for knowledge-driven assembly automation,” - *Autom.*, vol. 64, no. 3, pp. 231–243, 2016.
- [59] H. Cheng, P. Zeng, L. Xue, Z. Shi, P. Wang, and H. Yu, “Manufacturing Ontology Development Based on Industry 4.0 Demonstration Production Line,” in *2016 Third International Conference on Trustworthy Systems and their Applications (TSA)*, 2016, pp. 42–47.
- [60] B. R. Ferrer, W. M. Mohammed, A. Lobov, A. M. Galera, and J. L. M. Lastra, “Including human tasks as semantic resources in manufacturing ontology models,” in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 3466–3473.
- [61] “FASTory Simulator.” [Online]. Available: <http://escop.rd.tut.fi:3000/>. [Accessed: 17-May-2018].